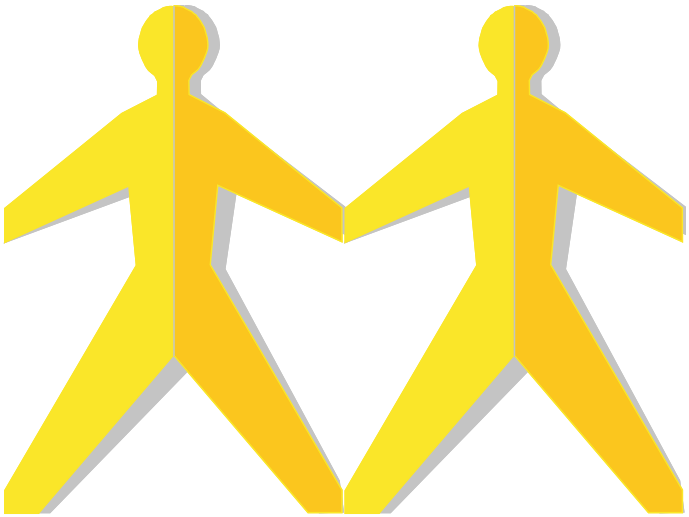
gedas



rvs® Tiny

Version 2.0

User Manual

The products listed in this manual are protected by copyright.

rvs® Tiny
User Manual

© 2004 by gedas deutschland GmbH
Pascalstraße 11
10587 Berlin

This manual is protected by copyright. All rights reserved. No part of this book may be used or reproduced in any form or by any means including photocopies, microfilm or any other means or stored in a database or retrieval system without obtaining prior permission from gedas. Rights are also reserved as far as lectures, radio and television is concerned.

We reserve the right to make changes to the content of this manual without giving prior notice. gedas is not liable for technical or printing errors or defects in this manual. Moreover, gedas shall not be liable for damage which is directly or indirectly caused by delivery, performance and use of this material.

Contents

1 Introduction 5

1.1 Short description of the system 5
1.2 Representation means 8
1.3 Target group 8

2 Installation on Windows systems 9

2.1 System requirements 9
2.2 Obtaining a license 9
2.3 Fresh install of rvs® Tiny 10
2.4 rvs® Tiny as Windows service 15
2.5 How to start rvs® Tiny GUI? 16
2.6 rvs® Tiny update installation 16

3 Configuration 17

3.1 Customizing the global configuration files 17
3.1.1 rvsTiny.properties 17
3.1.2 rvsTinyConfig.xml 18
3.2 Customizing the station configuration 19
3.2.1 Customizing stations via GUI 20
3.2.2 Customizing stations in the XML configuration file 23
3.2.3 Identification of rvs® Tiny stations 24
3.2.4 STATION_LOC station entry 26
3.2.5 STATION_NK station entry 26
3.2.6 STATION_RT station entry 28
3.3 Customizing the JobStarts 29

4 Working with rvs® Tiny 33

4.1 Starting the rvs® Tiny server 33
4.2 Stopping the rvs® Tiny server 33
4.3 Displaying Monitor messages 34
4.4 Activating a station 36
4.5 Sending a file 37
4.6 Synchronization of Send Jobs 43
4.7 Listing of all receive and send jobs 46
4.8 Deleting or releasing EERPs 50
4.9 Archiving the entries of processed send or receive jobs in the revision log 51

5 rvs® Tiny Central Administration 53

5.1 Introduction 53
5.2 Command Tools of the Central Administration 54
5.3 How to work with the central administration features? 56
5.3.1 How to exchange a license key file? 57
5.3.2 How to change a station parameter? 58
5.3.3 How to make an update of rvs® Tiny? 60
5.4 How to change from rvsWIN to rvs® Tiny? 61

Index 63

© gedas deutschland GmbH / Fachbereich rvs® Systems

1 Introduction

In this chapter you will find a short description of rvs® and rvs® Tiny as well as an explanation of typographic conventions used in the present manual.

1.1 Short description of the system

What is rvs®

rvs® = Rechner-Verbund-System

The abbreviation rvs® stands for the German word Rechner-Verbund-System. The rvs® computer communication system is a well established base service for electronic data interchange, EDI.

rvs® serves to ensure transmission of electronic data between heterogeneous computer platforms using different network protocols.

To do so, rvs® implements a universal network model, which you can configure in each network node.

rvs® provides an efficient and reliable transport service for both standardized EDI message types and files of any format or contents. You can receive only such files that are explicitly destined for rvs®. This means that rvs® does not allow any unauthorized access to remote or to own data files.

The system was originally developed by Volkswagen AG and has been used in the German and European automobile industries for a number of years but also by banks, insurances and industry worldwide.

rvs® uses the OFTP protocol. An extension to the OFTP standard was developed for Volkswagen AG: It has been enhanced by a line driver for SNA LU 6.2.

What rvs® is not

rvs® is not an online system. It neither supports direct terminal-like access to other sites, nor does it provide a communication pipe from application to application on a data record level. You cannot directly execute transfers in your own application. You rather can place send orders from within you application to rvs® which will be handled asynchronously.

rvs® is not a job scheduling system.

rvs® does not care about the contents of the files it is transporting. It only acts as a transparent transport medium and performs no semantic interpretation of the data it carries.

rvs® is not an EDI converter. You can, however, purchase additional components for converting between specific message formats (e.g. VDA, ODETTE, EDIFACT, XML) using rvs® as transport service from gedas deutschland GmbH.

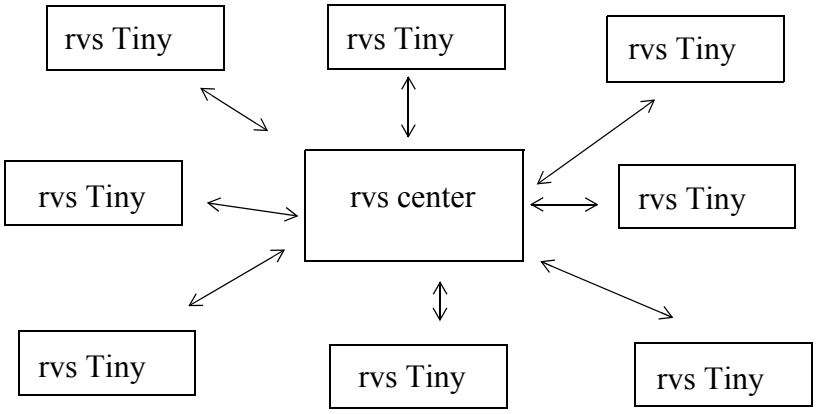
rvs® is not a network control or monitoring tool.

© gedas deutschland GmbH / Fachbereich rvs® Systems

What is rvs® Tiny

rvs® Tiny

rvs® Tiny is a communication software with a graphical user interface based, like rvs®, on the OFTP protocol. It can connect applications with partners. For this reason the software is ideal for star topologies where many branches or mobile employees are connected with one central server (see the following picture).



available features

The following features are available in rvs® Tiny 2.0:

- send files to the rvs® center or to the routed stations reachable via rvs® center.
- receive files from the rvs® center (or from routed stations via rvs® center).
- activate the rvs® center
- view information about receive jobs, send jobs, failed jobs and ended jobs.
- display monitor messages
- delete or release EERPs (End-to-End-Response) if necessary
- archive information about the processes send or receive jobs in the revision log.
- define job filters and actions when sending or receiving files
- code conversion with various code conversion tables
- format conversion
- log files for tracing the Monitor activities and for troubleshooting

rvs® Tiny uses a batch interface and the file system to communicate with the application. If capable to do so, the linked application can indicate successful processing and have successful dispatch indicated.

rvs® Tiny is implemented in Java.

TCP/IP protocol

At present, rvs® Tiny only supports the OFTP TCP/IP protocol.

Note: As rvs® Tiny is planned to substitute the product rvsWIN, there is a chapter 5.4 "How to change from rvsWIN to rvs® Tiny?" in this manual with some advices for the roll-out procedure.

Supported platforms

rvs® Tiny is a Java application and therefore supports a wide range of platforms. At the moment are the following installers availabe:

- rvs® Tiny for MS Windows XP with TCP/IP
- rvs® Tiny for MS Windows 2000 with TCP/IP

If you want an installer for another platform, please contact us:

rvs® after-sales service

phone: +49 30 39971 777

fax: +49 30 39971 994

email: rvs-service@gedas.de.

© gedas deutschland GmbH / Fachbereich rvs® Systems

1.2 Representation means

This chapter describes the typographic conventions used in this manual and explains the meaning of specially highlighted expressions.

Typographic conventions

- Instructions begin with a bullet.

– Other lists begin with the en dash.

Character styles

`Courier`	Commands, menu commands, file names, path names, programs, examples, scripts, options, qualifiers, data sets, fields, modes, window names, dialog boxes and statuses
BOLD and **IN CAPITAL LETTERS**	Parameters, environment variables, variables
“Inverted commas”	Links to other manuals, sections and chapters, literature
Bold	Important terms, names of operating systems, proper names, buttons, function keys.

Directories

$RVSTINY_HOME

As user directories are found on different locations for the different operating systems we use the variable **$RVSTINY_HOME** in this manual. Default values are:

– `C:\Programs\rvsTiny` for **Windows XP** and **Windows 2000**

Substitute the variable with your correct path.

1.3 Target group

This manual is meant for regular users of rvs® Tiny as well as administrators. It provides an overview of the basic rvs® Tiny functions.

Skills

The following skills are required to be able to use rvs® Tiny:

– good knowledge of the current operating system

– knowledge of the communications techniques in use TCP/IP.

Before starting to work with rvs® Tiny it is advisable to have read this book.

2 Installation on Windows systems

The present chapter describes the system requirements as well as the rvs® Tiny installation procedure.

2.1 System requirements

To successfully operate rvs® Tiny you need the following software:

Software

– Operating system: Windows XP or Windows 2000.
– Java runtime environment (JRE 1.3.1_XX or Java Software Development Kit 1.3.1_XX).

  Please make sure a Java runtime environment is present on your system prior to installing rvs® Tiny. The software is freely available for download from http://java.sun.com.

Initially, you need at least 8 MB free space on your hard disk. Depending on the amount of usage, the retention period for old entries, and the time between database cleanups, the space requirement may be considerably larger.

At present, rvs® Tiny only supports the OFTP TCP/IP protocol.

As a rule rvs® Tiny ships on CD-ROM or magtape. Your system must be able to read these media. Please contact your distributor if you have different requirements.

2.2 Obtaining a license

You need a license key to work with rvs® Tiny.

rvs® after-sales service

Please contact the rvs® after-sales service (phone: +49 30 39971 777; fax: +49 30 39971 994; email: rvs-service@gedas.de) to receive a license key.

To purchase a license key:

– Type `hostname` in the command prompt window (`Run -> cmd`).
– Send the command output (your computer name, e.g. `BWcd00034`) to the rvs® after-sales service. Please make sure to heed upper- and lowercases as otherwise a valid license key for your computer cannot be generated.
– You will be sent your license key file by email.

Save the license key file in the `$RVSTINY_HOME\conf\` directory as `license.properties`.

Note: Please read the chapter 1.2 for the explanation of `$RVSTINY_HOME.`

© gedas deutschland GmbH / Fachbereich rvs® Systems

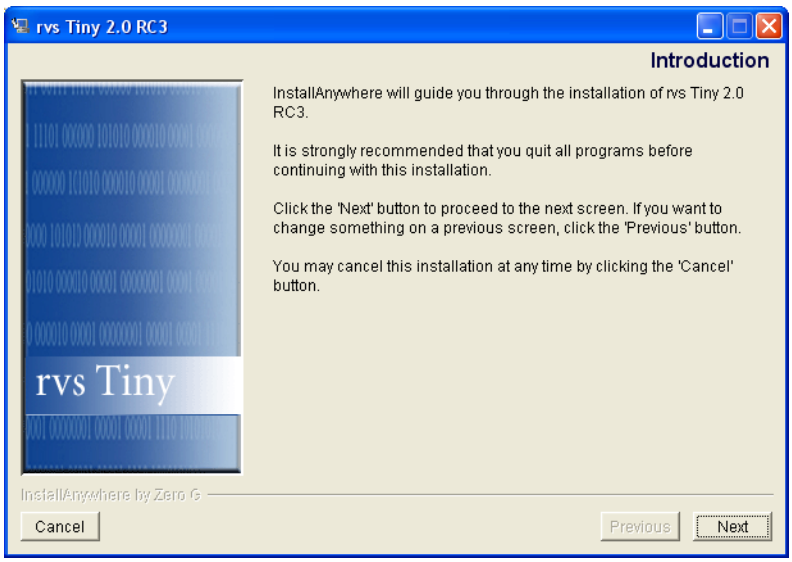
2.3 Fresh install of rvs® Tiny

Installation steps

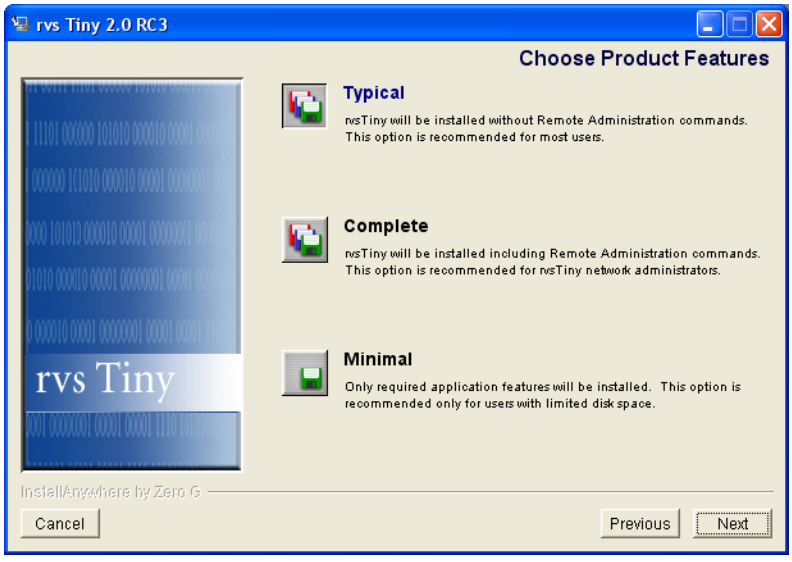
This chapter describes the installation of rvs® Tiny. Please read the chapter 2.1 „System requirements“ before installing.

- Start Windows and log in as a Windows user with administrator rights.
- Start the `rvsTiny_X.X_setup.exe` installer (where `X.X` indicates the rvs® Tiny version number) by double-clicking or using the Windows command: `Start -> Run`.
- Choose the installation language (German, English, Spanish…) in the first dialog. Click **<OK>** to go to the next installation step.

- The second dialog gives a brief introduction to the installation procedure.

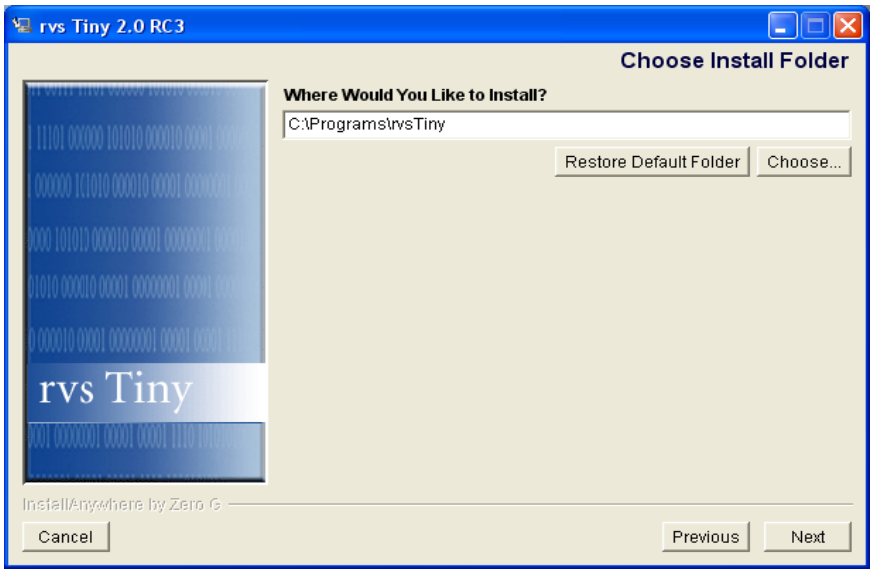

rvs Tiny 2.0 RC3
Introduction
InstallAnywhere will guide you through the installation of rvs Tiny 2.0 RC3.
It is strongly recommended that you quit all programs before continuing with this installation.
Click the 'Next' button to proceed to the next screen. If you want to change something on a previous screen, click the 'Previous' button.
You may cancel this installation at any time by clicking the 'Cancel' button.
rvs Tiny
InstallAnywhere by Zero G
Cancel
Previous
Next

- In the next dialog you can select between Typical, Complete or Minimal installation. Complete Installation is necessary, if you want to execute the central administration commands (such as update rvs® Tiny at another rvs® Tiny station, modify configuration of another rvs® Tiny stations,..).

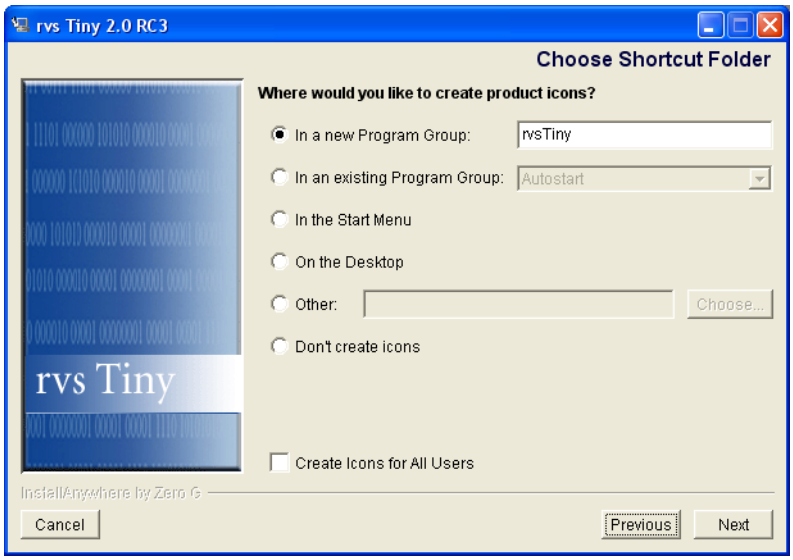

rvs Tiny 2.0 RC3
Choose Product Features
Typical
rvsTiny will be installed without Remote Administration commands.
This option is recommended for most users.
Complete
rvsTiny will be installed including Remote Administration commands.
This option is recommended for rvsTiny network administrators.
Minimal
Only required application features will be installed. This option is recommended only for users with limited disk space.
rvs Tiny
InstallAnywhere by Zero G
Cancel
Previous
Next

- Choose the rvs® Tiny installation directory in the third dialog.

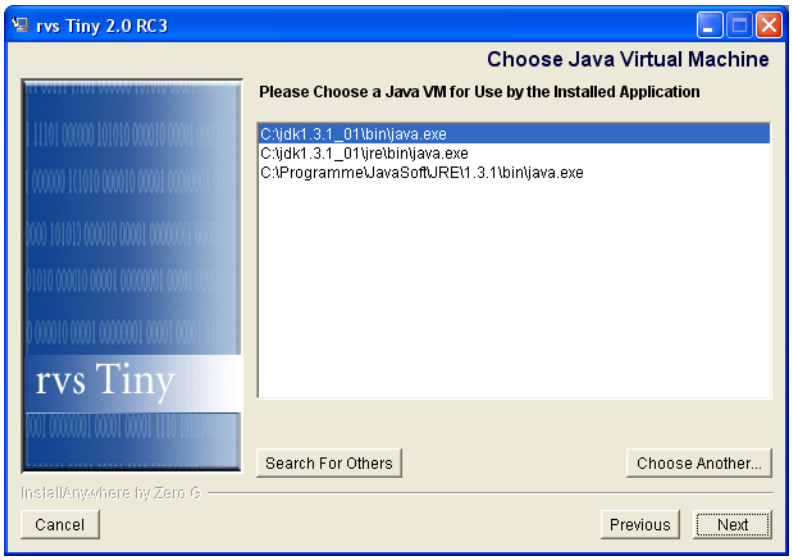
© gedas deutschland GmbH / Fachbereich rvs® Systems



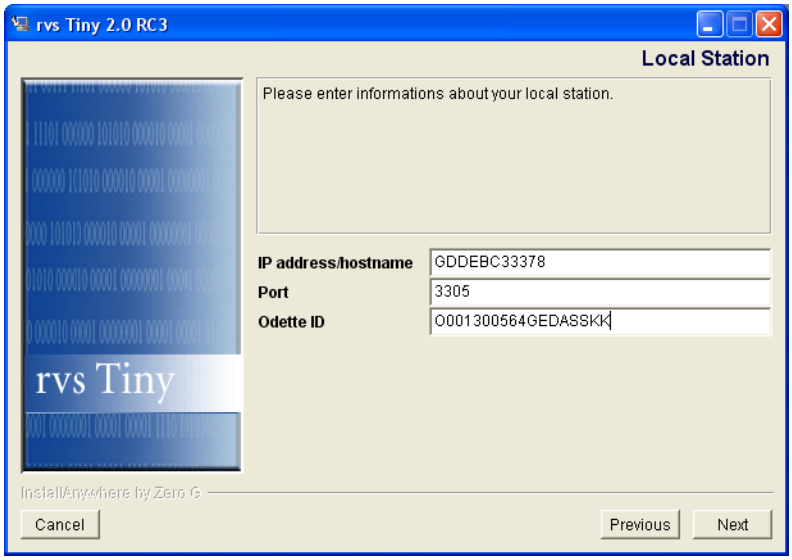
- Define the program group to install rvs® Tiny icons to in the fourth dialog.



- Define the Java runtime environment for rvs® Tiny operation in the next dialog. The installer searches for installed components and proposes the versions found in a dialog. rvs® Tiny has been released for use from version 1.3.1_X onwards.

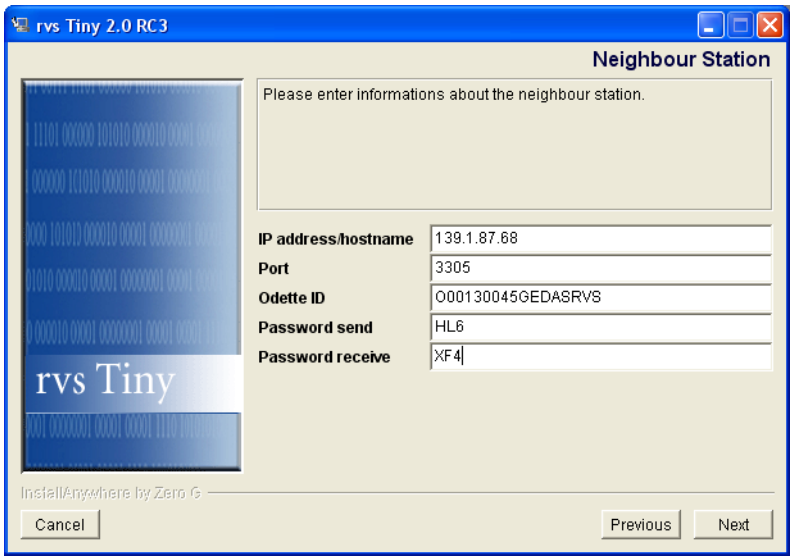


- The following dialogs deal with the settings in the station list. You must first configure the entry for the local station. The three parameters in the dialog are mandatory and must be specified. Please read the chapter 3.2 for more information about setting station parameter (e.g. how to obtain the ODETTE ID).



- Then follows the dialog for the direct neighbor station (rvs® center) parameters. All of these parameters are mandatory as well and must be specified to ensure proper operation of rvs® Tiny.

© gedas deutschland GmbH / Fachbereich rvs® Systems



- The last dialogs informs you of the successful installation of rvs® Tiny.

Starting and stopping

- To test, start and stop rvs® Tiny using the following commands:

  `startServer` / starts the rvs® Tiny process.

  `stopServer` / stops the rvs® Tiny process.

These two commands are available as batch files in the `$RVSTINY_HOME\bin` directory.

Starting and stopping via the command prompt:

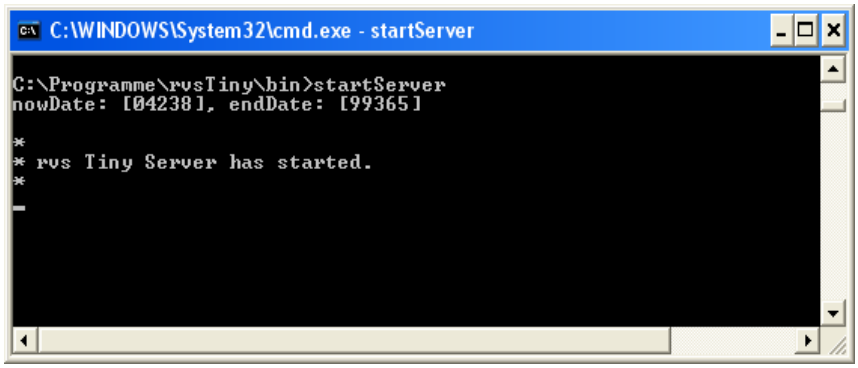
You can also start and stop rvs® Tiny using the command prompt: In this case you must change to the `$RVSTINY_HOME\bin` directory (e.g. `C:\Programs\rvsTiny\bin`).

Successful start:

Start

A successful start is indicated as follows:

:



Starting and stopping via the Start menu:

Start rvs® Tiny by choosing `Start -> Programs -> rvsTiny -> rvsTiny` (or the Start menu command you specified during installation) in the Start menu. This will guide you to the `$RVSTINY_HOME\bin` directory where you can start any of the (`startServer.bat`; `stoperver.bat`) batch files by double-clicking.

2.4 rvs® Tiny as Windows service

Unter some circumstances it is desirable to run rvs® Tiny in background even though you are not logged in to the computer. Therefore you have to install rvs® Tiny as a system service.

Note: The term Service means a program that can be started from the operating system and works in the background.

This is possible starting the batch script `rvstinyservice.bat` out of the directory `$RVSTINY_HOME\bin`.

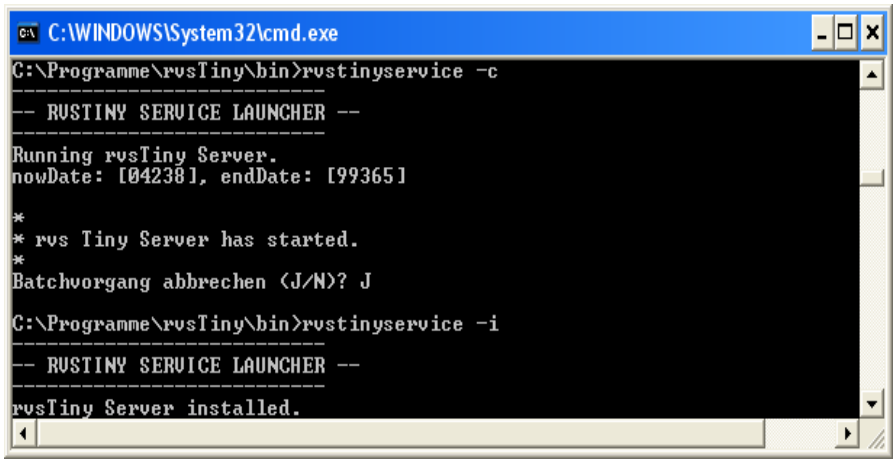
Usage:

```
rvstinyservice <parameters> [options]
```

.The possible parameters are:

-c	starts rvs® Tiny on console
-i	installs rvs® Tiny as service
-r	removes rvs® Tiny as service
-s	starts rvs® Tiny as service
-h	usage

Example:



© gedas deutschland GmbH / Fachbereich rvs® Systems

With the command `rvstinyservice -c` rvs® Tiny Server was started on the command line (console).

With the command `rvstinyservice -i` rvs® Tiny was installed as system (Windows) service.

Now you can find rvs® Tiny as a service in the list of system services (`Start - > Control Panel -> Administrative Tools -> Services`). If you want to start rvs® Tiny every time the system starts, you can set the startup type to `Automatic` by choosing Automatic from the combo box in the **Startup type** area.

Example (Windows XP german version):

2.5 How to start rvs® Tiny GUI?

To start the GUI use the program `startGUI`. It is a batch file in the directory `$RVSTINY_HOME\bin.`

So, you have to change to the `$RVSTINY_HOME\bin` directory (e.g. `C:\Programs\rvsTiny\bin`) to be able to execute this program.

Alternatively, you can start rvs® Tiny GUI using the rvs® Tiny program group (`Start -> Programs -> rvsTiny -> Start GUI`).

Usage:

```
startGUI [-local]
```

All parameters are optional.

-local	starts the OFTP server module in the same process as the GUI, no transmission of data will be possible without the GUI running, an already running OFTP server process might lead to errors.
-help	Requests help information.
-?	Requests help information.

Example: `startGUI`

Please read the chapter 3 for more information about the features of rvs® Tiny GUI.

2.6 rvs® Tiny update installation

Please read the chapter 5 for more information how to update rvs® Tiny with central administration.

3 Configuration

The present chapter describes how to customize configuration files via the rvs® Tiny GUI or via rvs® Tiny command line.

Note: You do not need to make any changes to obtain a functioning system if you have correctly specified all parameters during installation and you do not require more stations than the local station and the rvs® center. You must restart rvs® Tiny each time you have made changes to a configuration file.

3.1 Customizing the global configuration files

Files Two configuration files are of vital significance for the global rvs® Tiny settings.

- `$RVSTINY_HOME\conf\rvsTinyConfig.xml`
- `$RVSTINY_HOME\conf\rvsTiny.properties.`

3.1.1 rvsTiny.properties

This is a Java properties file containing entries of the following format:

`<name>=<value>.`

This file refers to 3 important locations (contains three important parameters):

- The rvs® Tiny installation directory (parameter: `RootDir`)
- The global rvs® Tiny configuration file (parameter: `ConfigFile`)
- The global configuration file for system logging (parameter: `LogConfigFile`).

Sample file (comment lines starting with # will be ignored):

```
RootDir=C:\\Programs\\rvsTiny
# configuration xml file
ConfigFile=C:\\Programs\\rvsTiny\\conf\\rvsTinyConfig.xml
# log config file
LogConfigFile=C:\\Programs\\rvsTiny\\conf\\rvsTinyLogger.xml
```


© gedas deutschland GmbH / Fachbereich rvs® Systems

3.1.2 rvsTinyConfig.xml

Elements

The following table lists the most important elements of the `rvsTinyConfig.xml` file.

ELEMENT	DESCRIPTION
DB	Directory for job administration with the `ENDED`, `FAILED`, `RCV` and `SND` subdirectories. The `RCV` and `SND` subdirectories are used to store temporary, not fully processed jobs. The `FAILED` directory holds the failed, the `ENDED` directory the completed jobs. These directories are also visible in the GUI (Admin window, Jobs).
TEMP	Directory for temporary use.
INBOX	Directory where completely received files are stored.
OUTBOX	Temporary directory for files to be sent.
ARCDIR	Archive directory where files such as the Revision log are stored.
LOGDIR	Directory for both log files: `monlog.log` and `tiny.log`.
JobstartConfigFile	Configuration file for job start.
StationsConfigFile	Station configuration file: Comprises the configuration parameters for the local station, the neighbor station, and routed stations.
HostAllowFile	Configuration file containing DNS names or IP addresses of hosts that may send rvs® Tiny commands to the rvs® Tiny server.
HostDenyFile	Configuration file containing DNS names or IP addresses of hosts that may not send rvs® Tiny commands to the rvs® Tiny server.

Example:

Excerpt from `rvsTinyConfig.xml`

```
...
<Environment>
<DB>c:\Programs\rvsTiny\jobs</DB>
<TEMP>c:\Programs\rvsTiny\files\temp</TEMP>
<INBOX>c:\Programs\rvsTiny\files\inbox</INBOX>
<OUTBOX>c:\Programs\rvsTiny\files\outbox</OUTBOX>
<ARCDIR>c:\Programs\rvsTiny\archive</ARCDIR>
<JobstartConfigFile>rvsTinyJobstart.xml</JobstartConfigFile>
<StationsConfigFile>rvsTinyStationlist.xml</StationsConfigFile>
<MonlogStylesheet>MonlogStylesheet.xslt</MonlogStylesheet>
<PersistenceArchive>RevisionLog.xml</PersistenceArchive>
<HostAllowFile>host.allow</HostAllowFile>
<HostDenyFile>host.deny</HostDenyFile>
<RMIServiceName>rvsTiny</RMIServiceName>
```

```
<RMIServiceHost>localhost</RMIServiceHost>
<JobQueueTimeout>3000</JobQueueTimeout>
</Environment>
...
```

########## rvsTinyConfig.xml ##################

You can define and edit the paths for the DB, TEMP, INBOX, OUTBOX, LOGDIR, and ARCDIR elements.

Example:

```
<DB>D:\rvsTiny\jobs</DB>
<TEMP>C:\Programs\rvsTiny\temp</TEMP>
```

You are free to choose the names for the rvsTinyJobstart and rvsTinyStationlist files; the only requirement is that they are specified in the respective XML element, are valid XML files and are located in the conf directory. The same also applies to HostAllowFile and HostDenyFile.

Example:

```
<StationsConfigFile>stationen.xml</StationsConfigFile>
```

The stationen.xml file is a station list in XML format containing the required rvs® Tiny parameters (see chapter 3.2) and is located in the rvs® Tiny conf directory.
The entries for RMISeviceName and RMIServiceHost are for internal communication and must not be changed.

Note: You must stop and restart rvs® Tiny each time you have made changes to any configuration file (see chapter 4.1 and 4.2).

3.2 Customizing the station configuration

Station Configuration

The installer offers you the possibilty to set up a local station and the rvs® center.

For further customization of stations (creation of new stations and their modification or deletion) you can use:

– the Graphical User Interface (GUI) or

– configuration files in XML format.

The obligatory parameters for these two stations (the local and the rvs® center) have already been polled during installation, and the

$RVSTINY_HOME\conf\rvsTinyStationlist.xml

station configuration file has been appropriately adapted. This configuration is also visible in the GUI. If you do not need more entries for routed stations, this completes the station configuration; sending and receiving files is now possible.

© gedas deutschland GmbH / Fachbereich rvs® Systems

3.2.1 Customizing stations via GUI

To start the GUI use the program `startGUI`.

Usage:

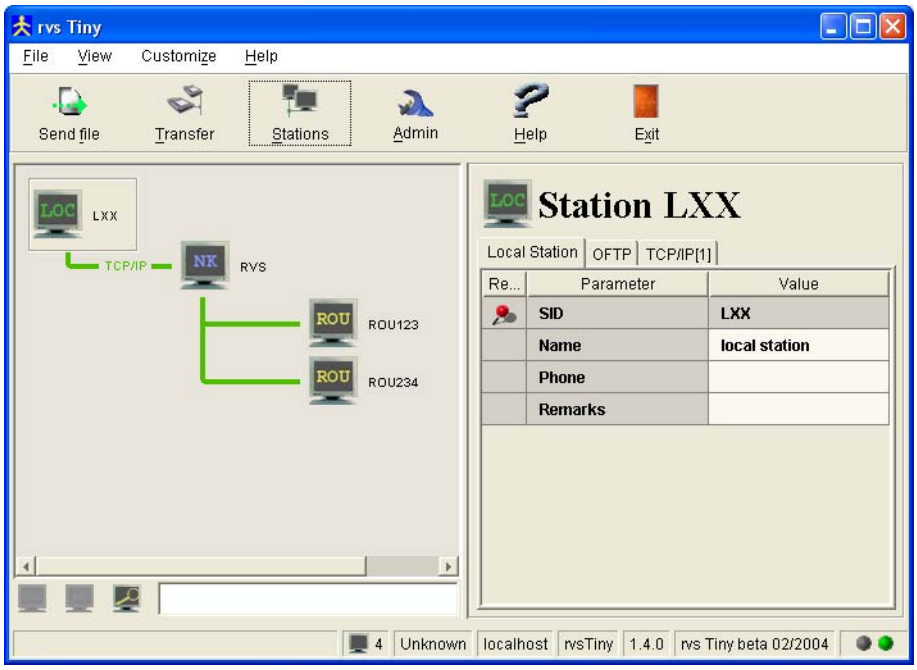
```
startGUI [-local]
```

All parameters are optional.

-local	starts the OFTP server module in the same process as the GUI, no transmission of data will be possible without the GUI running, an already running OFTP server process might lead to errors.
-help	Requests help information.
-?	Requests help information.

After the successfull start of the GUI the following window appears.

To open the station window, select the Stations icon in the function bar.



On the left side of the stations window you will see the station tree; to the right is a station parameter table.

The station tree depicts all of the stations which exist in the rvs® Tiny database (e.g. your local station and the rvs® center, you have already configured during the installation, so they are now visible in the station tree) as well as their connection type (TCP/IP at present).

The station table on the right-hand side of the window displays all of the parameters for the station currently selected. With the aid of the various station tabs (OFTP, TCP/IP) you can configure various parameter groups.

Grayed fields indicate that these parameters cannot be edited.

Note: Once you have saved the parameters for a station, you can no longer change the station ID (SID). Changing the station ID would mean configuring an entirely new station.

The parameters which are obligatory for station configuration are marked in the column **Re**.. (Required) with the symbol .

Example: `Odette Id` is obligatory in the OFTP tab.

On the right-hand side of the window beneath the station parameter table is a series of buttons: Save, Cancel, Undo, Undo All. These allow you to save changes (Save), discard them (Cancel) or reverse them (Undo, Undo All).

Configurating a local station

The possible tabs for the local station configuration are: Local Station, OFTP (Odette parameters), TCP/IP. Please read the chapter 3.2.3 and 3.2.4 for detailed explanation of those groups of parameters. The parameters `Name`, `Phone` and `Remarks` from the tab Local Station are comments. The tab TCP/IP is

© gedas deutschland GmbH / Fachbereich rvs® Systems

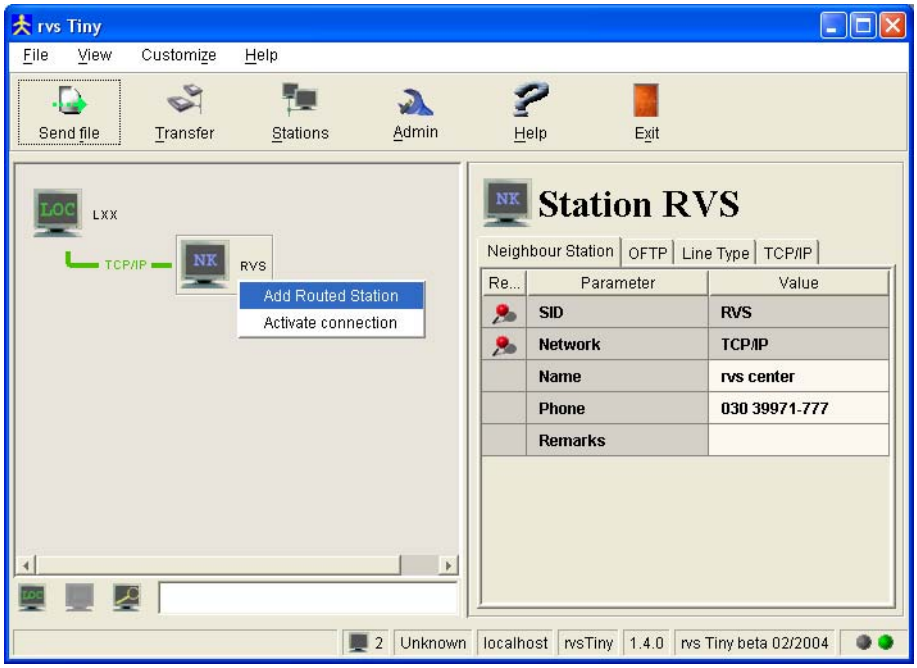
equivalent to the TCPIP_BASICS element (parameter group) in the XML station configuration file.

Configuration of the rvs® center

The possible tabs for the rvs® center are: Neigbour Station, OFTP (Odette parameters), Line Type, TCP/IP. Please read the chapter 3.2.3 and 3.2.5 and for detailed explanation of those groups of parameters.

A right-click on the rvs® center opens the context menu, which offers then the options **Add routed station** and **Activate connection**. With the option **Add routed station** you can add a set of routed stations, that are reachable via the rvs® center. **Activate connection** activates the connection to the rvs® center.

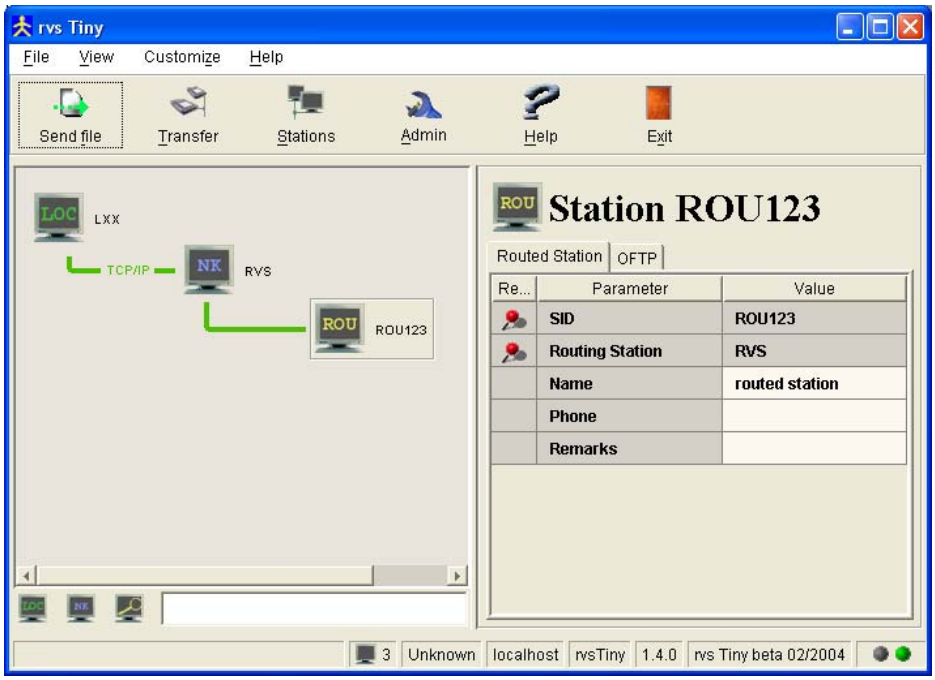
Setting up a routed station



Pre-condition: You must already have set up a direct neighbouring node, the rvs® center, via which you can reach the routed station.

Set up a routed station by a right-click on the neighbour station (the rvs® center) and selecting from the context menu the item **Add routed station**. For you, the connection type (line type) by which this station is to be reached is of no importance (this is dealt with by the direct neightbouring node, the rvs® center). For this reason, the connection type to the routed station is not shown.

The possible tabs for this type of station are: Routed Station, OFTP (Odette parameters). Please read the chapter 3.2.3 and 3.2.6 for detailed explanation of those groups of parameters.



A routed station in rvs® Tiny can only be reached via the rvs® center. This is shown by the parameter Routing Station in the GUI that is equivalent to the parameter GATEWAY_STATION_NK in the element (parameter group) STATION_RT of the XML station configuration file.

Besides entering the freely selectable station name (SID) in the Routed Station tab, you only have to enter the `Odette ID` in the Odette tab to complete the station configuration of the routed station.

Once you have saved the parameters for a routed station, you can no longer change the station ID (SID). Changing the station ID would mean configuring an entirely new station without deleting the old one.

A right-click on the routed station opens the context menu, which offers then an option **Delete station**.

3.2.2 Customizing stations in the XML configuration file

On the other hand the station configuration can be done editing the XML station configuration file `rvsTinyStationlist.xml` (see chapter 3.1.2 for the explanation, where this file is to be found). The STATION_LOC element in the configuration file is equivalent to the Local Station in the GUI, STATION_NK is Neighbour Station and STATION_RT is Routed Station in the GUI.

Excerpt from the `rvsTinyStationlist.xml` file:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<RVS_STATION_CONFIG>
<!-- Setting for the local station -->
<STATION_LOC>
 . . .
</STATION_LOC>
<!-- Setting for the neighbour station -->
<STATION_NK>
```


© gedas deutschland GmbH / Fachbereich rvs® Systems

```
 . . .
<STATION_NK>
<!-- Setting for routed stations -->
<!-- One entry for each routed station -->
<STATION_RT>
. . .
</STATION_RT>
</RVS_STATION_CONFIG>
```

Comments

Lines starting with (<!--) and ending with (-->) are interpreted as comments.

Edit this file if you have to change mandatory parameters (**ODETTE_ID**, **TCPIP_BASIC**) or if you wish to assign values to other optional parameters such as **CONTACT**.

Use a text editor (e.g. `Edit`, `TextPad`) to do so. Please make sure to save your XML files as valid XML documents after editing as rvs® Tiny otherwise will not be able to read them and may fail to start correctly.

The changes in the GUI will be visible immediately (afer saving) in the XML station configuration file; the changes in the XML station configuration file will be visible in the GUI only after a new start of the rvs® Tiny (command `startServer`).

Note: For correct TCP/IP communication you must ensure that the IP ports for RMI (1099) and for Odette (e.g. `3305`) are free.

The following chapters will describe all station parameters in detail.

3.2.3 Identification of rvs® Tiny stations

Station list

There are several parameters in the station list that identify rvs® Tiny stations. Of these only the **SID**, **ODETTE_ID** and **TCPIP_BASIC** parameters are obligatory. The SID parameter was assigned during rvs® Tiny installation; the parameter for the local station is `LOC`, and `RVS` for the rvs® center.

SID	This is a locally unique station ID which must consist of up to sixteen characters. It is a strictly local definition; remote stations do not have access to these names; they only know the ODETTE IDs.
ODETTE_ID	ODETTE ID is a worldwide unique identification of all nodes using the ODETTE file transfer protocol (OFTP). This 25 character name consists of – the letter O, – an 18 character organization identifier provided by the ODETTE codification group, and – a 6 character computer sub address that is administrated by each organization. If you communicate within your own closed network only, the ODETTE ID may be freely chosen as long as it remains unique in your network.
CONTACT	Contact information to be further specified later; these parameters are optional.

../NAME	Name of the contact partner responsible for this station (computer).
../ENTERPRISE	Company/enterprise
../LOCATION	Location
../STREET	Street
../STREETNUMBER	Street number
../DEPARTMENT	Department
../PHONE	Phone number
../FAX	Fax number
../MAIL	Email address
../INET	Enterprise website
../FREETEXT	At your disposal.
TCPIP_BASIC	TCP/IP parameters required for connection establishment.
../IP_ADDR	IP address or DNS name of the own or the partner station.
../PORT	Port on which a TCP/IP listener is to be started; 3305 by default.
../SND_PW	The password rvs® Tiny sends to the neighbor station. ODETTE password exchange between two neighbor stations and verification always occurs while a session is being established.
../REC_PW	The password rvs® Tiny expects from the neighbor station.
../EERP_OUT	Procedure for sending receipts (EERPs). `NORMAL`: Generation of a receipt after successful file reception and immediate active transmission. `HOLD`: Generation of a receipt after successful file reception. The receipt, however, is only sent after having been released with the `handleEERP` program. Default: `NORMAL`

At present, the station list comprises the following, not yet evaluated parameters:
- The **<TCPIP_REC>** block with the **<MAX_INCOMING_SESSIONS>** and **<MAX_OUTGOING_SESSIONS>** parameters;
- The **ENABLED** parameter.

To apply for an ODETTE ID in Germany, please contact:

ODETTE ID

Verband der Automobilindustrie e.V. (VDA)
Abt. Logistik
Postfach 17 05 63
60079 Frankfurt
Ph.: +49 69-7570-0

© gedas deutschland GmbH / Fachbereich rvs® Systems

Get the complete description of OFTP from:

`http://www.odette.org/`

3.2.4 STATION_LOC station entry

Local station

This area comprises information on the local station; it must be present exactly once in the station configuration.

Local station example:

```
<STATION_LOC>
   <SID>LOC</SID>
   <ODETTE_ID>O0013005623GEDASSKK</ODETTE_ID>
   <CONTACT>
      <NAME>local station</NAME>
      <ENTERPRISE/>
      <LOCATION/>
      <STREET/>
      <STREET_NUMBER/>
      <DEPARTMENT/>
      <PHONE> </PHONE>
      <FAX/>
      <MAIL/>
      <INET/>
      <FREETEXT></FREETEXT>
   </CONTACT>
   <TCPIP_BASIC>
      <IP_ADDR>139.1.87.68</IP_ADDR>
      <PORT>3305</PORT>
   </TCPIP_BASIC>
   <TCPIP_REC>
      <MAX_INCOMING_SESSIONS>5</MAX_INCOMING_SESSIONS>
      <MAX_INCOMING_SESSIONS>1</MAX_INCOMING_SESSIONS>
   </TCPIP_REC>
   <ENABLED>Yes</ENABLED>
</STATION_LOC>
```

In the following example, the local station **SID** is `LOC`, the **ODETTE ID** `O0013005623GEDASSKK`. The neighbor node can reach this station under IP address `139.1.87.68` and at port `3305`.

TCPIP_REC

Note: The **TCPIP_REC** block with the **MAX_INCOMING_SESSIONS** and **MAX_OUTGOING_SESSIONS** parameters and the **ENABLED** parameter are not being evaluated at present and are reserved for future application. Do not edit nor delete these parameters.

3.2.5 STATION_NK station entry

rvs® center

This area contains information on a direct neighbor station, rvs® center only. Again this area must be present in the station configuration exactly once only.

```
<STATION_NK>
   <SID>RVS</SID>
   <ODETTE_ID>O0013005623GEDASMEL</ODETTE_ID>
   <CONTACT>
```

```
      <NAME>neighbour station</NAME>
      <ENTERPRISE/>
      <LOCATION/>
      <STREET/>
      <STREET_NUMBER/>
      <DEPARTMENT/>
      <PHONE> </PHONE>
      <FAX/>
      <MAIL/>
      <INET/>
      <FREETEXT></FREETEXT>
   </CONTACT>
   <LINE_TYPE>
      <LINE_SUSP>No</LINE_SUSP>
      <PSESSIONS>-1</PSESSIONS>
      <ACTIVE_CON_SETUP>Yes</ACTIVE_CON_SETUP>
      <DELAY>0</DELAY>
   </LINETYPE>
   <ODETTE>
      <SND_PW>H5C</SND_PW>
      <REC_PW>L3Y</REC_PW>
      <SND_BLOCKS>0</SND_BLOCKS>
      <REC_BLOCKS>0</REC_BLOCKS>
      <EX_BUF_CRE>0</EX_BUF_CRE>
      <EX_BUF_SIZ>0</EX_BUF_SIZ>
      <EERP_IN>NORMAL</EERP_OUT>
      <EERP_OUT>NORMAL</EERP_OUT>
   </ODETTE>
   <TCPIP_BASIC>
      <IP_ADDR>139.1.65.42</IP_ADDR>
      <PORT>3305</PORT>
   </TCPIP_BASIC>
</STATION_NK>
```

SID	Station ID (see chapter 3.2.3).
ODETTE_ID	ODETTE ID (see chapter 3.2.3).
CONTACT	Contact information (see chapter 3.2.3).
LINE_TYPE	For future application, not being evaluated.
ODETTE	Parameter block, comprises parameters for OFTP.
../SND_PW	The password rvs® Tiny sends to the neighbor. ODETTE password exchange between two neighbor stations and verification always occurs while a session is being established.
../REC_PW	The password rvs® Tiny expects from the neighbor.
../EX_BUF_CRE	For future application, not being evaluated.
../EX_BUF_SIZ	For future application, not being evaluated.

© gedas deutschland GmbH / Fachbereich rvs® Systems

../EERP_OUT	Procedure for sending receipts (EERPs). `NORMAL`: Generation of a receipt after successful file reception and immediate active transmission. `HOLD`: Generation of a receipt after successful file reception. The receipt, however, is only sent after having been released with the `handleEERP` program. Default: `NORMAL`
TCPIP_BASIC	TCP/IP parameters required for connection establishment.
../IP_ADDR	IP address or DNS name of the own or the partner station.
../PORT	Port on which a TCP/IP listener is to be started; 3305 by default.

3.2.6 STATION_RT station entry

Routing station

This entry defines for each rvs® station through which neighbors it can be reached.

```
<STATION_RT>
   <SID>ROUTEDSTATION</SID>
   <ODETTE_ID>OROUT</ODETTE_ID>
   <GATEWAY_STATION_NK>RVS</GATEWAY_STATION_NK>
   <CONTACT>
      <NAME/>
      <ENTERPRISE/>
      <LOCATION/>
      <STREET/>
      <STREET_NUMBER/>
      <DEPARTMENT/>
      <PHONE/>
      <FAX/>
      <MAIL/>
      <INET/>
      <FREETEXT/>
   </CONTACT>
   <ODETTE>
      <EERP_OUT>NORMAL</EERP_OUT>
   </ODETTE>
</STATION_RT>
```

SID	Station ID (see chapter 3.2.3).
ODETTE_ID	ODETTE ID (see chapter 3.2.3).
CONTACT	Contact information (see chapter 3.2.3).
GATEWAY_STATION_NK	This field comprises the station's SID used to execute communication with the routed station configured here. For rvs® Tiny is may only be the direct neighbor station (rvs® center) because rvs® Tiny only allows for a direct neighbor station.
ODETTE	Parameter block, comprises parameters for OFTP.

../EERP_OUT	Procedure for sending receipts (EERPs). `NORMAL`: Generation of a receipt after successful file reception and immediate active transmission. `HOLD`: Generation of a receipt after successful file reception. The receipt, however, is only sent after having been released with the `handleEERP` program. Default: `NORMAL`
TCPIP_BASIC	TCP/IP parameters required for connection establishment.
../IP_ADDR	IP address or DNS name of the own or the partner station.
../PORT	Port on which a TCP/IP listener is to be started; 3305 by default.

3.3 Customizing the JobStarts

The JobStart configuration comprises rules that allow special programs to be launched when appropriate files are being sent or received.

JobFilter

All programs of the jobFilters in question are started (their sequence cannot be influenced) if more than one jobFilter applies to the send or receive job.

It is possible to customize JobStarts via the GUI or via the JobStart configuration file.

Customizing via GUI

At first you should open the Administration window selecting the Admin icon in the function bar. Then select the item Jobstart in the Administration tree on the left size. How to start GUI read please in the chapter 2.5.

It is possible to chosse between Jobstarts in receive direction and Jobstarts in send direction. A Jobstart in receive direction is equivalent to a resident receive entry in rvs®. A Jobstart in send direction is equivalent to a Jobstart after send attempt in rvs®.

A new Jobstart will be created with a right-click on a Jobstart after receive or a Jobstart after send attempt in the Administration tree (**Add new entry**). To select the already existing Jobstart, double-click the appropriate line of the appropriate JobStart in the right-hand window.

The following JobFilters' entries are possible: Direction, SID, VDSN, Process and SendAttempts. Please refer to the table of the JobFilters for a detailed description.

Customizing via XML configuration file `rvsTinyJobstart.xml`

Like most of the other rvs® Tiny configuration files this file is in the XML format as well.

```
<jobstarterData>
       <jobfilters>
              <jobfilter>
                     <vdsn></vdsn>
                     <sid></sid>
```

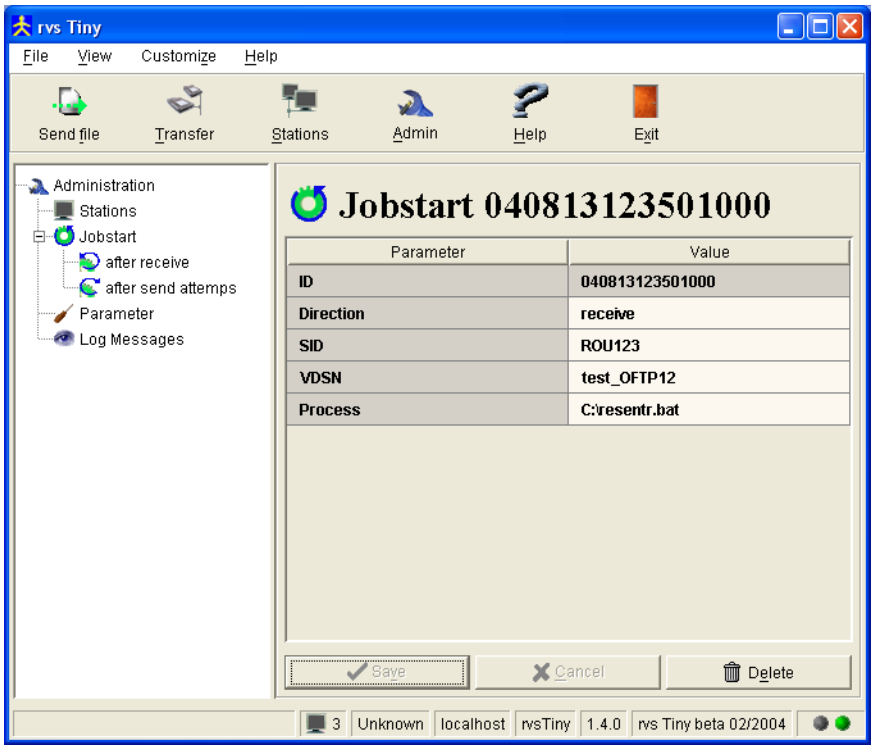

© gedas deutschland GmbH / Fachbereich rvs® Systems

```
                    <direction>SND</direction>
                    <sendAttempts>0</sendAttempts>
                    <process>C:\jobstart.bat</process>
                </jobfilter>
                <jobfilter>
                 ...
                </jobfilter>
                ...
        </jobfilters>
</jobstarterData>
```

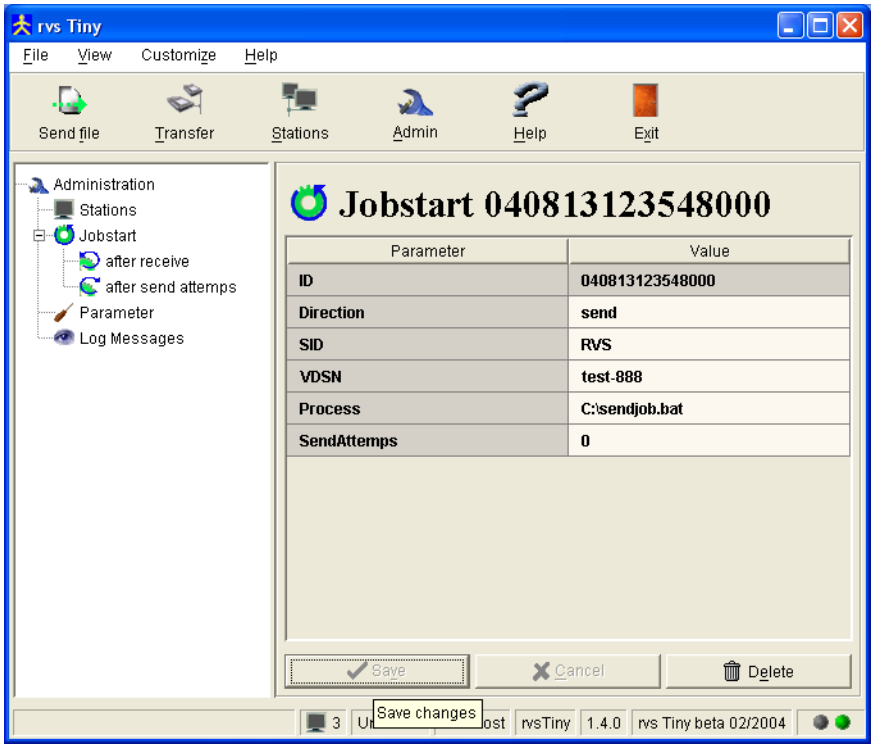
This file comprises any number of jobFilter elements. The table below gives a detailed description of individual jobFilter sub elements.

vdsn	Virtual file name (regular expression as filter).
sid	Station ID (regular expression as filter for the station ID).
direction	Defines filter rules for the communication direction. Possible values: `SND` (when sending files), `RCV` (when receiving files).
sendAttempts	Number of failed attempts to send. Successful file transmission is indicated by "0" here.
process (mandatory)	Program to be started when all filter conditions apply. A defined set of parameters is passed to the programs. Parameters: 1. jobId 2. Station ID (of sender or recipient) 3. File name of the file sent or received 4. VDSN 5. Date and time of the job 6. Number of attempts to send.

Example (GUI) If rvs® Tiny receives a file with the virtual file name `test_OFTP12` from a routed station `ROU123` the program `C:\resentr.bat` will be started.



Example (GUI) If rvs® Tiny sends successfully (`SendAttempts=0`) a file with the virtual file name `test-888` to the rvs® center `RVS`, the program `C:\sendjob.bat` will be started.



© gedas deutschland GmbH / Fachbereich rvs® Systems

Delete a JobStart

If you want to delete a JobStart, select it at first with a double-click. Now you can remove it with the **Delete** button.

4 Working with rvs® Tiny

Batch files

The present chapter describes all programs available for everyday use of rvs® Tiny. These programs are located as batch files in the `$RVSTINY_HOME\bin` directory or available via the rvs® Tiny GUI.

Note: To launch an rvs® Tiny program you must change to the `$RVSTINY_HOME\bin` directory.

4.1 Starting the rvs® Tiny server

`startServer`

Use the `startServer` program to start rvs® Tiny.

Example:

```
startServer
```

It is not possible to specify any parameters.

A successful start is indicated as follows:

```
*
* rvs Tiny Server has started.
*
```

Note: The RMI port 1099 is necessary for the RMI registry, so if this port is occupied, rvs® Tiny Server will not start successfully.

4.2 Stopping the rvs® Tiny server

`stopServer`

Use the `stopServer` program to stop rvs® Tiny.

Usage:

```
stopServer -m <mode> [-verbose]
```

All parameters are optional.

-m <mode>	Time for jobs to terminate before rvs® Tiny is stopped. Possible values: `0` (default; 120 seconds), `1` (60 seconds), 2 (30 seconds), `3` (20 seconds), `4` (10 seconds),
-verbose	Verbose message output.
-help	Requests help information.

© gedas deutschland GmbH / Fachbereich rvs® Systems

-? Requests help information.

Example:

```
stopServer
```

Result: The server stops after 120 seconds.

Example:

```
stopServer -m 3
```

Result: The server stops after 20 seconds.

```
*
* rvs Tiny Server has stopped.
*
```

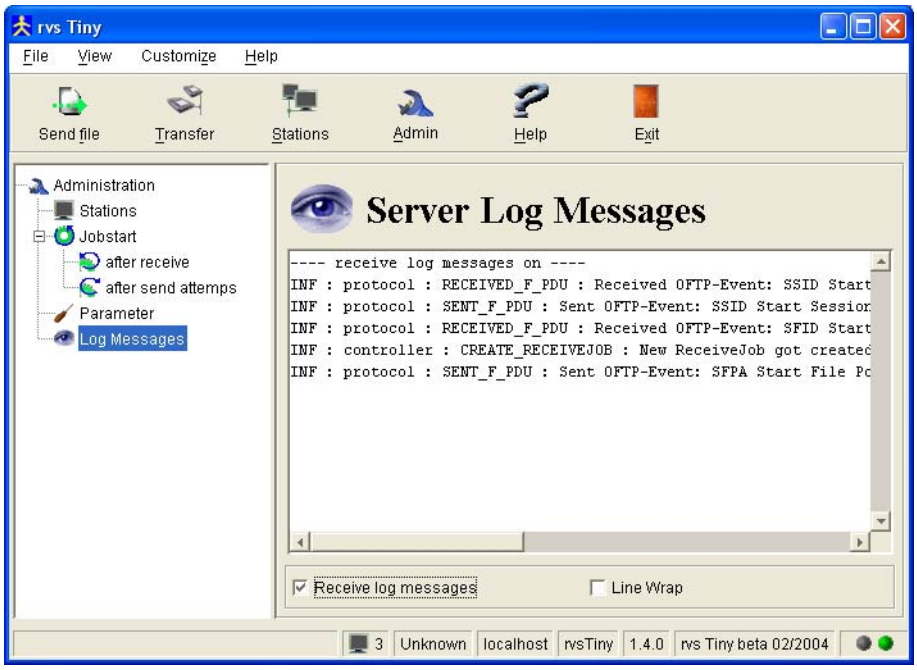
4.3 Displaying Monitor messages

It is possible to display Monitor messages via the GUI or via the command line. Use this program to trace the current Monitor messages and to analyze error messages if applicable.

GUI

Select the Admin icon in the function bar of rvs® Tiny GUI . The Administration window opens with the Admin tree and the sub-entry Log Messages. How to start of rvs® Tiny GUI please read the chapter 2.5.

Placing a tick in the box marked **Receive log messages** allows you to view the messages from the `$RVSTINY_HOME\rvsTiny\log\monlog.log` file. Only messages occuring after the GUI is started will be displayed.



Command Line

Use the `showMonitorLog` program to trace the current Monitor messages and to analyze error messages if applicable.

Usage:

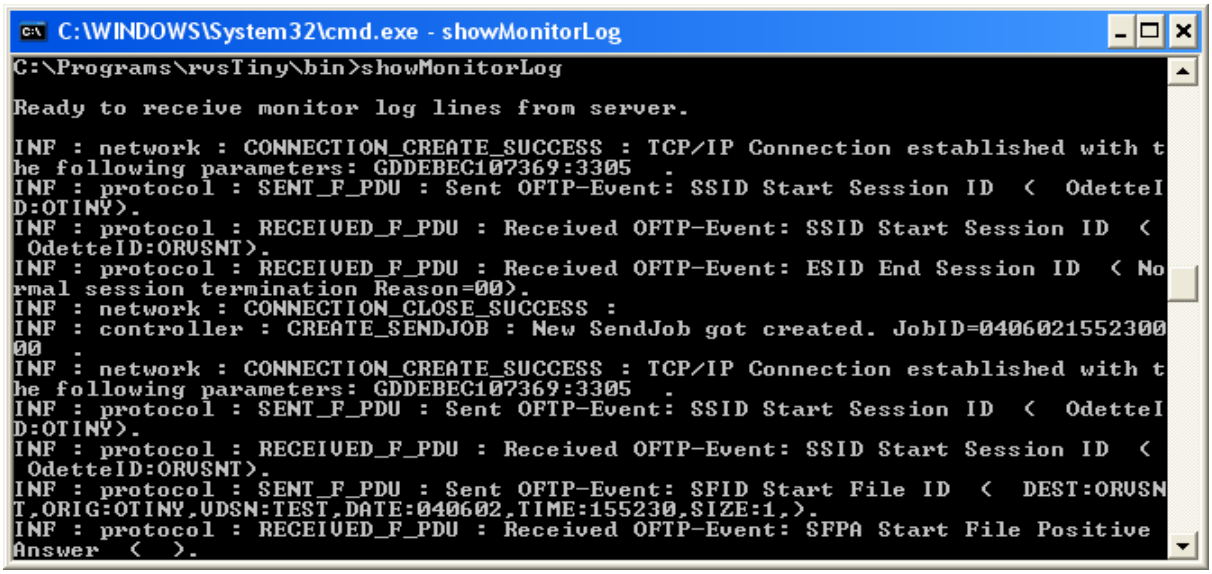
```
showMonitorLog [-verbose]
```

Optional parameters:

-verbose	Verbose message output.
-help	Displays a description of the current command
-?	Requests help information.

© gedas deutschland GmbH / Fachbereich rvs® Systems

Example:


C:\WINDOWS\System32\cmd.exe - showMonitorLog
C:\Programs\rvsTiny\bin>showMonitorLog
Ready to receive monitor log lines from server.
INF : network : CONNECTION_CREATE_SUCCESS : TCP/IP Connection established with the following parameters: GDDEBEC107369:3305 .
INF : protocol : SENT_F_PDU : Sent OFTP-Event: SSID Start Session ID (OdetteID:OTINY).
INF : protocol : RECEIVED_F_PDU : Received OFTP-Event: SSID Start Session ID (OdetteID:ORVSNT).
INF : protocol : RECEIVED_F_PDU : Received OFTP-Event: ESID End Session ID (Normal session termination Reason=00).
INF : network : CONNECTION_CLOSE_SUCCESS :
INF : controller : CREATE_SENDJOB : New SendJob got created. JobID=040602155230000 .
INF : network : CONNECTION_CREATE_SUCCESS : TCP/IP Connection established with the following parameters: GDDEBEC107369:3305 .
INF : protocol : SENT_F_PDU : Sent OFTP-Event: SSID Start Session ID (OdetteID:OTINY).
INF : protocol : RECEIVED_F_PDU : Received OFTP-Event: SSID Start Session ID (OdetteID:ORVSNT).
INF : protocol : SENT_F_PDU : Sent OFTP-Event: SFID Start File ID (DEST:ORVSNT,ORIG:OTINY,VDSN:TEST,DATE:040602,TIME:155230,SIZE:1,).
INF : protocol : RECEIVED_F_PDU : Received OFTP-Event: SFPA Start File Positive Answer ( ).

The Monitor messages are written to the `$RVSTINY_HOME \rvsTiny\log\monlog.log` file.

4.4 Activating a station

`activate Station`

Use the `activateStation` program to activate the rvs® center in the command line. How to activate the rvs® center via the GUI, please see chapter 3.2.1.

Note: You cannot activate a routed station. You can activate only a direct neighbouring node (neighbour station, rvs® center).

Usage:

```
activateStation [-verbose]
```

Optional parameters:

-verbose	Verbose message output.
-help	Displays a description of the current command.
-?	Requests help information.

Heed messages in the command prompt window starting with message or error if rvs® center activation fails (e.g. due to an incorrect IP address).

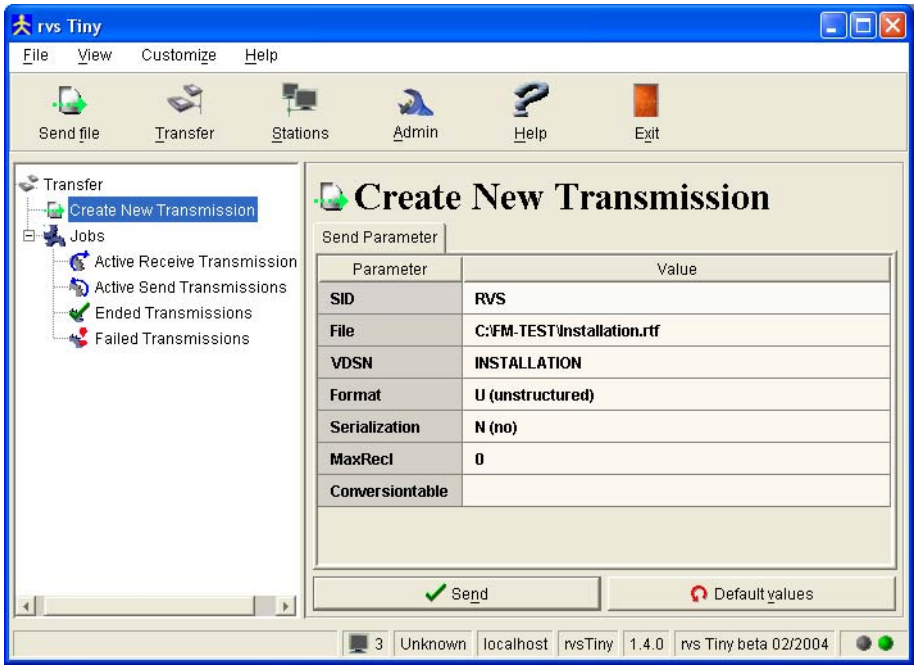
Correct the configuration file with the station table when the IP address is incorrect. You must stop and restart rvs® Tiny each time you have saved the configuration file.

4.5 Sending a file

It is possible to send a file via the GUI or via the command line. How to start GUI, please read the chapter 2.5.

GUI

Select the icon **Send file** or **Transfer** in the function bar. The Transfer window opens with the sub-entry Create New Transmission in the Transfer tree. Selection of Create New Transmission opens a window on the right-hand size of the screen entitled Create New Transmission. Now you can type or select the send parameters.



Required parameters:

SID	Recipient station ID.
File	File name of the file to be sent.
VDSN	Virtual file name; the length of the file name used for ODETTE transfer must not exceed 26 characters.

© gedas deutschland GmbH / Fachbereich rvs® Systems

Optional send parameters:

Format	Format of the file to be sent: – T=text file; a stream of ASCII characters – U=unstructured (binary); byte stream – V=variable; variable record length – F=fixed; fixed record length The files in format F or V must be text files with the appropriate line (record) length. **Example**: If you want to send a file in format fixed with the record length 80, it must be a text file, where each line has a length 80 without CR/LF. The following parameters are to be set: Format=F, MaxRecl=80.
Serialization	Y (Yes)/N (No). If you set Serialization=Y, the files will be sent in the same order, as the send jobs were created. The next job will only be sent, if the previous is completely finished. All send jobs for the serialization must have the same LABEL. In the GUI the VDSN will be used as label. That means, that all jobs for the same serialization group must have the same virtual data set name (VDSN).
MaxRecl	maximal record length for the files in format F or V, please see the examle for the parameter Format.
Conversion table	For ASCII - EBCDIC conversion the following conversion tables are available : ASCII-IBM037, ASCII-IBM237, ANSI-IBM073, ANSI-IBM273. For EBCDIC - ASCII conversion the following conversion tables are available : IBM037-ASCII, IBM237-ASCII, IBM037-ANSI, IBM273-ANSI. Please see the note beneth the parameters' table for the command line for more explanation about the conversion tables.
-h	Requests help information.
-verbose	Verbose message output.
-?	Requests help information.

Command Line

`createSendJob` Use the `createSendJob` program to send a file to the rvs® center via the command line.

Usage:

```
createSendJob -d <filename> -s <receiver sid>
[-I <input code> -O <output code>
 -t <conversion table>]
[-F <format>] [-M <length>]
[-S <serialisation> -l <label>][-h?][-verbose]
```

Required parameters:

-d <filename>	File name of the file to be sent.
-s <receiver sid>	Recipient station ID.
-v <vdsn>	Virtual file name; the length of the file name used for ODETTE transfer must not exceed 26 characters.

Optional send parameters:

-F <format>	Format of the file to be sent: – T=text file; a stream of ASCII characters – U=unstructured (binary); byte stream – V (variable); variable record length – F (fixed); fixed record length. The files in format F or V must be text files with the appropriate line (record) length. **Example**: If you want to send a file in format fixed with the record length 80, it must be a text file, where each line has a length 80 without CR/LF. The following parameters are to be set: `-F=F -M=80`
-M <length>	maximal record length for the files in format F or V, please see the examle for the parameter Format.

© gedas deutschland GmbH / Fachbereich rvs® Systems

-I <input code>	With the parameters input code and output code, you can select one of the conversion tables, that rvs® Tiny offers. The possible conversion tables are: For ASCII - EBCDIC conversion the following conversion tables are available : ASCII-IBM037, ASCII-IBM237, ANSI-IBM073, ANSI-IBM273. For EBCDIC - ASCII code conversion the following conversion tables are available : IBM037-ASCII, IBM237-ASCII, IBM037-ANSI, IBM273-ANSI. Please, see the note beneath this table for details about code tables. **Example**: If you want to use the conversion table ASCII-IBM237, as input code you should write ASCII (`-I ASCII`); as output code IBM237 (`-O IBM237)` . It is also possibile to set as input/output code only the values A/E. In this case is A ASCII and E IBM037 (code conversion ASCII-IBM037).
-O <output code>	see input code.
-t <conversion table>	your own conversion table with the complete path. Please read the User Manual, chapter Code Conversion for more information how to create the own conversion table.
-S <serialize>	Y (Yes)/N (No). If you set the option `-S=Y`, the files will be sent in the same order as the send jobs were created. The next job will only be sent, if the previous is completely finished. All send jobs for the serialization must have the same LABEL. In the GUI the VDSN will be used as label. In the command line you can specify your own label.
-l <label>	Name of group of serialized send jobs. User specified (descriptive) label for this job. If you do not specify this parameter the VDSN will be user as a label.
-h	Requests help information.
-verbose	Verbose message output.
-?	Requests help information.

Note: The text file are stored on most systems in one of two computer codes, namely ASCII (American National Standard Code for Information Interchange) or EBCDIC (Extended Binary Coded Decimal Interchange Code).

ASCII is the standard code for UNIX and DOS/Windows systems. Here is the short expanation of the conversion tables, that are offered by rvs® Tiny.

EBCDIC was developed for IBM Mainframe computers.

- **ASCII**: US-ASCII ISO 646; the ASCII character set defines 128 characters (0 to 127 decimal). This character set is a subset of many other character sets with 256 characters, including the ANSI character set of MS Windows.
- **ANSI**: Windows ANSI, Values 0 to 127 are the same as in the ASCII character set, values 128 to 255 are similar to the ISO Latin-1 character set.
- **EBCDIC 037**: support characters, which are used in the following countries: Australien, Brasilien, Kanada, Neuseeland, Portugal, Südafrika, USA.
- **EBCDIC 273**: supports characters (especially umlauts), which are used in the following countries: Germany, Austria and Switzerland.

Examples:

```
createSendJob -d C:\text.txt -s RVS -v test
```

In this example the `C:\text.txt` file is sent to the `RVS` rvs® center with a virtual name `test33`. The virtual file name parameter (option `-v`) is mandatory.

Examples:

```
createSendJob -d C:\text.txt -s RVS -v OFTP_TEST
-F F -M 80 -I ANSI -O IBM273
```

In this example the `C:\text.txt` file is sent to the rvs® center `RVS` with a virtual name `OFTP_TEST`, this file is a text file of whicheach line has a length of 80 characters without CR/LF (`-F F -M 80`). Before transmission the file will be converted from ANSI to IBM037 (`-I ANSI -O IBM273`) code.

```
createSendJob -d C:\TEMP\part.txt -s RVS -v PART
-S Y -l AUTO
```

In this example the `C:\TEMP\part1.txt` file is sent to the rvs® center `RVS` with a virtual name `PART`, this file belongs to the serialized group of files with the label (for the whole group) `AUTO`.

The message `createSendJob exited with return code 0` appears when a send job was successfully created.

A temporary file named, for example, `040329170027000` is created in the `SND` directory while a send job is being processed. This name is made up of the date (`040329`), the time (`170027`) and a consecutive three-digit number (`000`) for files arriving in the very same second.

In the following example job `040329170027000` waits for a receipt (**EERP**) from the rvs® center and is therefore still in the `SND` directory. `SND` and `RCV` are directories where jobs currently being processed are intermediately stored. The Status field indicates the job's processing phase. `WF_EERP` (**W**aiting **F**or **E**ERP) means: waiting for EERP (receipt).

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```


© gedas deutschland GmbH / Fachbereich rvs® Systems

```
<Job>
       <ID>040329170027000</ID>
       <Static>
              <FileName>C:\Answer.txt</FileName>
              <VDSN>TESTFILE</VDSN>
              <SID>RVS</SID>
              <Direction>SND</Direction>
              <Date>040329</Date>
              <Time>170027</Time>
       </Static>
       <Dynamic>
              <RestartPos>0</RestartPos>
              <RecCount>0</RecCount>
              <FilePos>19</FilePos>
                 <RcvBytes>0</RcvBytes>
                 <SendAttempts>0</SendAttempts>
                 <Status>WF_EERP</Status>
        </Dynamic>
</Job>
```

rvs® Tiny copies the job into the `ENDED` directory if it has been successfully terminated (e.g. after an `EERP` has been received). The `FAILED` directory holds jobs that could not be successfully processed.

Status Possible states for Send Jobs:

- RESTART=1 (wait after error to send file again).
- WF_SFID_ANSWER=2 (wait for answer for already sent SFID).
- WF_CDT=3 (wait for credit renewal).
- WF_EFID_ANSWER=4 (wait for answer to already sent EFID).
- WF_EERP=5 (wait for EERP).
- ENDED=6 (send job has ended).
- ENDED_WITH_JS_ERROR=7 (send job run into error during call of jobstart).
- FATAL_ERROR=8 (send job run into fatal error).

Possible states for Receive Jobs:

- RESTART (wait after error to receive file again).
- RESTART_AFTER_EFNA=2 (File could not be delivered; EFNA sent or not. Assuming partner will send file again).
- RESTART_AFTER_EFPA_FAILURE=3 (File already delivered, but EFPA failed. Assuming partner will send file again.).
- RECEIVING=4 (Receiving file data after EFPA).
- EERP_HOLDED=5 (File was completely received. EERP is on hold and needs to get released.).
- EERP_RELEASED=6 (User released EERP, but it is still not sent).
- EERP_DELETED=7 (User deleted EERP. Job will stop).
- ENDED=8 (Receive job has successfully ended after sending EERP.)

- ENDED_WITH_JS_ERROR=9 (Receive job run into error during calling jobstart).
- FATAL_ERROR=10 (Receive job run into fatal error).

Note: Please see in the rvs® portable, Reference Manual (chapter 3.3) for ODETTE protocol sequences (such as EFNA, SFID, EFID, EERP, ...).

4.6 Synchronization of Send Jobs

The Odette file transfer is asynchronous. That means: If you create a send job, a file will be only provided to be sent. The sending of the file will not possible, only after a connection to a partner station has been established.

In automated business with a lot of processes is desirable to react directly, if the sending of a file was not successfilly in a certain period of time.

As a solution for this problem rvs® Tiny offers on client-side a functionality for synchronized file transfer. This program remains active, until the transfer is successfully finished or an error occurs. It is possible to define the number of send attempts or the time in which a file fransfer has to be successfully done, or the whole transfer will be count as an error.

Note: A transfer is regarded as having been successfully completed when the ODETTE acknowledgement EERP (End-to-End Response) for this transfer has been received.

The `convertAndSend` program offers you the synchronization of send jobs functionality. It is similar to the `createSendJob program`: it has additionally the options for the synchronized file transfer and for the conversion of an EDI message with the EDI converter WEDIConv (see WEDIConv User Manual).

Usage:

```
convertAndSend -d <filename> -s <receiver sid>
[-I <input code> -O <output code> -t <conversion
table>] [-F <format>] [-M <length>] [-S
<serialisation> -l <label>] [-za <attempts>] [-zt
<timeout>] [converter parms]

[-h?][-verbose]
```

Required parameters:

-d <filename>	File name of the file to be sent.
-s <receiver sid>	Recipient station ID.
-v <vdsn>	Virtual file name; the length of the file name used for ODETTE transfer must not exceed 26 characters.

© gedas deutschland GmbH / Fachbereich rvs® Systems

Optional send parameters:

-F <format>

Format of the file to be sent:

- T=text file; a stream of ASCII characters
- U=unstructured (binary); byte stream
- V (variable); variable record length
- F (fixed); fixed record length.

The files in format F or V must be text files with the appropriate line (record) length.

Example: If you want to send a file in format fixed with the record length 80, it must be a text file, where each line has a length 80 without CR/LF. The following parameters are to be set: `-F=F -M=80`

-M <length>

maximal record length for the files in format F or V, please see the examle for the parameter Format.

-I <input code>

With the parameters input code and output code, you can select one of the conversion tables, that rvs® Tiny offers. The possible conversion tables are:

For ASCII - EBCDIC conversion the following conversion tables are available : ASCII-IBM037, ASCII-IBM237, ANSI-IBM073, ANSI-IBM273.

For EBCDIC - ASCII code conversion the following conversion tables are available : IBM037-ASCII, IBM237-ASCII, IBM037-ANSI, IBM273-ANSI.

Please, see the note beneath this table for details about code tables.

Example: If you want to use the conversion table ASCII-IBM237, as input code you should write ASCII (`-I ASCII`); as output code IBM237 (`-O IBM237)` .

It is also possibile to set as input/output code only the values `A`/`E`. In this case is `A` ASCII and `E` IBM037 (code conversion ASCII-IBM037).

-O <output code>

see input code.

-t <conversion table>

your own conversion table with the complete path. Please read the User Manual (chapter Code Conversion) for more information how to create the own conversion table.

-S <serialize>	Y (Yes)/N (No). If you set the option -S=Y, the files will be sent in the same order as the send jobs were created. The next job will only be sent, if the previous is completely finished. All send jobs for the serialization must have the same LABEL. In the GUI the VDSN will be used as label. In the command line you can specify your own label.
-l <label>	Name of group of serialized send jobs. User specified (descriptive) label for this job. If you do not specify this parameter the VDSN will be used as a label.
-za <attempts>	count of send attempts for synchronized transmission.
-zt <timeout>	time-out for synchronized transmission in seconds

Optional convert parameters:

-cf	format description for converter step 1
-cf2	format description for converter step 2 (optional)
-ct	stylesheet for converter step 1, optional
-cd	converter direction for converter step 1, default: EDI2XML
-cd2	converter direction for converter step 2, default: XML2EDI.
-cl	log level for converter, default: 0
-cs	line feed behaviour for the EDI-message output; 0 - no LF as segment separator, 1 - LF as segment separator (default).
-cI	identation for XML output; 0 - no identation; 1 - identation on (default).
-ce	encoding for the XML output; default: UTF-8.

Optional parameter:

-verbose

-help

-?

© gedas deutschland GmbH / Fachbereich rvs® Systems

Examples:

```
convertAndSend -d C:\teil56.txt -s RVS -v TEILE
-za 4
```

In this example the `C:\teil56.txt` file is sent to the rvs® center `RVS` with a virtual name `TEILE`, count of send attempts is limited to 4.

```
convertAndSend
-d C:\INTEGRATION\test.txt -s RVS -v TEST
-cf C:\rvsET\system\fmtDesc\fw.kanban.ineas.xml
-ct C:\rvsET\system\stylesheets\ineas2deljit.xslt
-cf2 C:\rvsET\system\fmtDesc\edifact.97.orig.xml
```

In this example the `C:\INTEGRATION\test.txt` file is sent to the rvs® center `RVS` with a virtual name `TEST`. The `test.txt` file was also converted with the EDI converter **WEDIConv** (installed in the directory `C:\rvsET\`); first from the inhouse format `test.txt` to the XML format (`fw.kanban.ineas.xml`) and then to the EDIFACT message (`edifact.97.orig.xml`). In the first step was used a stylesheet `ineas2deljit.xslt` for the special presentation.

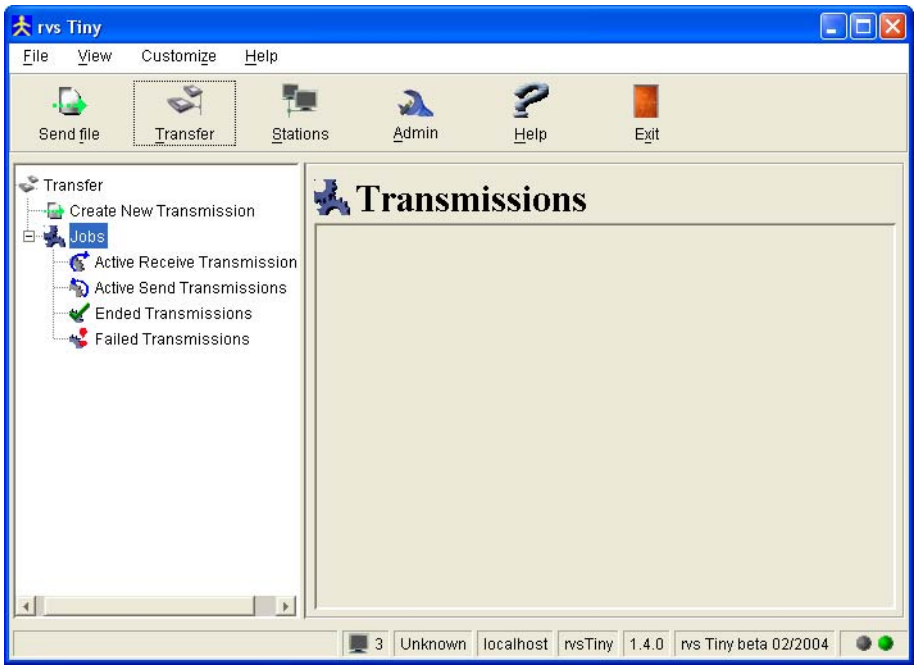
Note: If you use `convertAndSend` to send files, you must pay attention to the fact, that with this feature a new send job after an occured error can cause a double transmission (e.g. if the file is already transmitted, but the in the `-zt` parameter defined time out period passed without receiving the EERP. The transmission can be still active independent from the own local station (e.g. by the rvs® center). In this case, the new sending of the file can cause double transmission. The application above rvs® must be able to handle this particular situation. A possible solution is: the file name must be unique or it must get a unique counter (a counter stamp)).

4.7 Listing of all receive and send jobs

The most important information about the send and receive jobs can be shown via the GUI or via the command line. How to start GUI, please read the chapter 2.5.

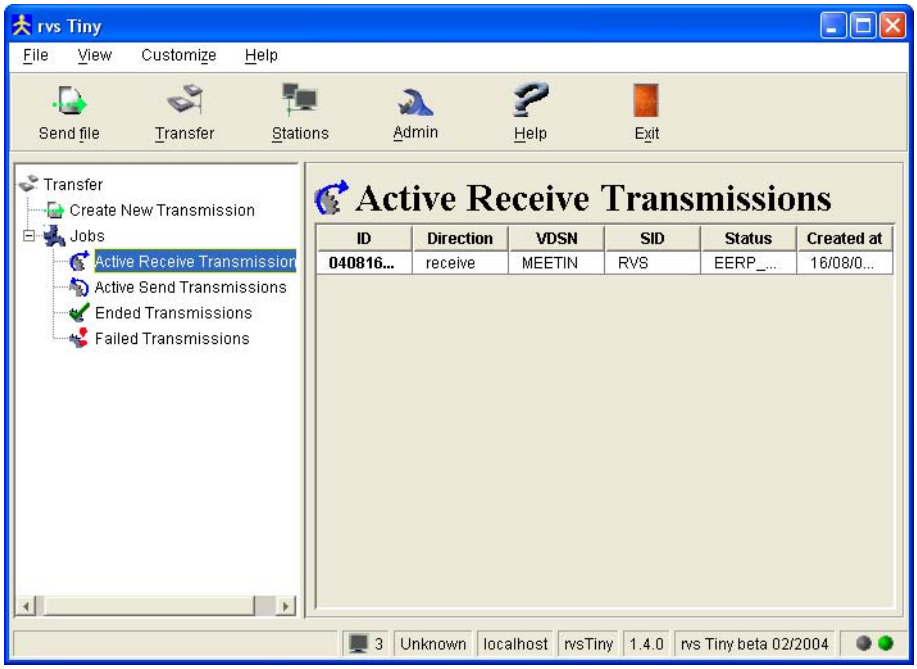
GUI

Select the icon Transfer in the function bar. The Transfer window opens with the sub-entry Jobs in the Transfer tree. Now you can select between the Active Receive/Send Transmissions, Ended Transmission and Failed Transmissions.



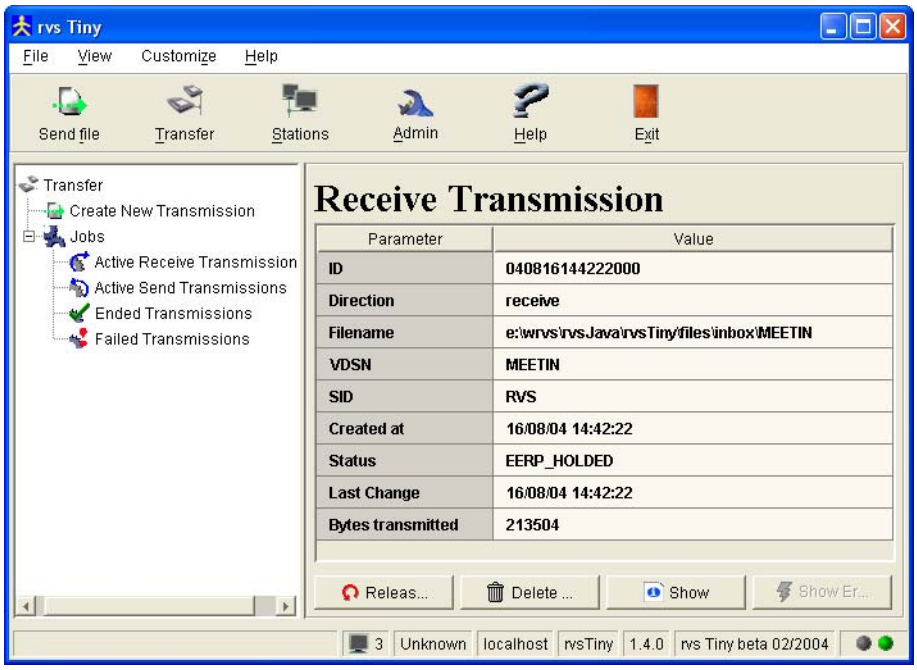
Note: A transfer is regarded as having been successfully completed when the ODETTE acknowledgement EERP (End-to-End Response) for this transfer has been received.

Single-click on selected job sub-ordner gives you an overview of all jobs in this ordner in the right-hand window.



By double-clicking on a job line in the right-hand section of the window you can obtain a detailed view of the relevant job.

© gedas deutschland GmbH / Fachbereich rvs® Systems



Active Receive Transmissions: If you click on the line Active Receive Transmission and then double-click on a particular job line, you will see all details about the particular receive job in transmission.

Release EERP_OUT

With the buttons in this right-hand window you can change the status of the EERP_OUT for this particular job. The Button **Release** releases the EERP_OUT in status Hold (status=EERP_HOLDED).

Delete EERP_OUT

The Button **Delete** deletes the EERP_OUT. The same functionality is available by the program `handleEERP` (see chapter 4.8).

Delete/Release Job

Active Send Transmissions: To delete this job, activate the **Delete** button. If the job is active, you must first pause it, as rvs® Tiny will not allow you to delete a job which is actually being transmitted. To pause the job, use the button marked **Hold**. A job, which has been paused can be released again by pressing the **Release** button.

Command Line

`getJobList`

Use the `getJobList` program to list all jobs.

Usage:

```
getJobList [-a][-e][-f][-verbose]
```

Optional parameters:

-a	Detailed information on jobs currently being processed.
-e	Information on terminated jobs.

-f	Information on failed jobs.
-h	Requests help information.
-verbose	Verbose message output.
-?	Requests help information.

Example:

```
getJobList -e
```

Result:

Information on a job entry

`getJob` Use the `getJob` program to retrieve information on a particular send or receive job.

Usage:

```
getJob -n <jobid> [-a] [-verbose]
```

Required parameters:

-n <jobid>	Information on a send or receive job with ID `<jobid>`.

Optional parameters:

-a	All available job information is given.
-help	Requests help information.
-verbose	Verbose message output.
-?	Requests help information.

Example:

```
getJob -n 040329173456000
```

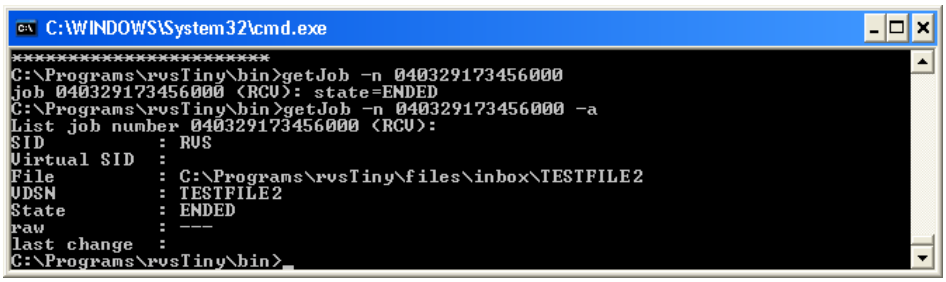

© gedas deutschland GmbH / Fachbereich rvs® Systems

Result: `job 040329173456000 (RCV): state: ENDED`

Example:

```
getJob -n 040329173456000
```

Result:



4.8 Deleting or releasing EERPs

`handleEERP` Use the `handleEERP` program to delete or release receipts (`EERP_OUT`). It is also possible to delete or release EERPs via the GUI (Please see the chapter 4.7).

Usage:

```
handleEERP -r|-d <num> [-verbose]
```

Required parameters:

-r | -d <num>	ID of the job for which the EERP is to be released/ deleted.

Optional parameters:

-help	Requests help information.
-verbose	Verbose message output.
-?	Requests help information.

4.9 Archiving the entries of processed send or receive jobs in the revision log

`archiveJobs` The `archiveJobs` program creates a `revision.log` file in the `$RVSTINY_HOME\rvsTiny\archive` directory; this file is used to log successfully processed send or receive jobs.

Usage:

```
archiveJobs [-verbose] [-help] [-?]
```

Optional parameters:

-help	Displays a description of the current command.
-verbose	Verbose message output.
-?	Requests help information.

Terminated jobs are deleted from the `$RVSTINY_HOME\rvsTiny\jobs\ENDED` directory and the data is written into the `revision.log` file in the `$RVSTINY_HOME\rvsTiny\archive` directory.

© gedas deutschland GmbH / Fachbereich rvs® Systems

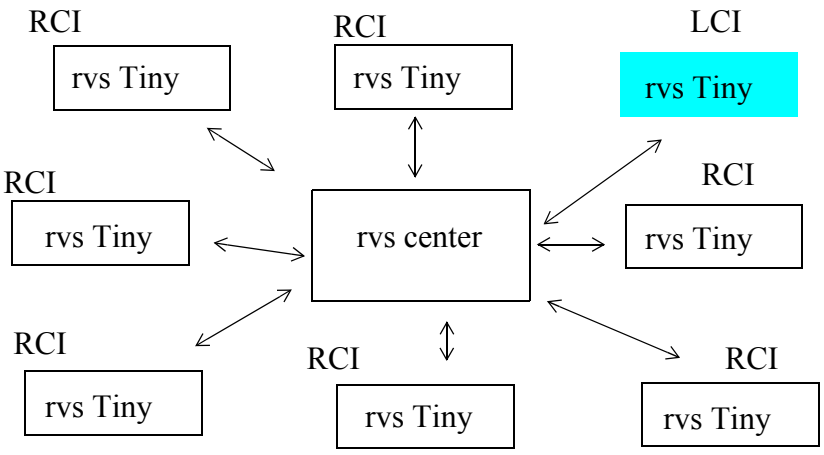
5 rvs® Tiny Central Administration

This chapter describes the powerful feature of rvs® Tiny: how to remotely administer other rvs® Tiny stations. rvs® Tiny is an ideal software for star topologies, where many stations are connected with one central server (rvs® center). Therefore it is essential for the network administrator being able to remotely administer all rvs® Tiny installations.

5.1 Introduction

The rvs® Tiny Central Administration enables the configuration of numerous rvs® Tiny installations by using special configuration files.

The configuration files will be sent from a rvs® Tiny station (we call it: Local Configuration Instance LCI) to the station, that should be administrated (we call it: Remote Controlled Instance RCI). The both instances are located in the rvs® Tiny star network (Please see the following picture).



To distinguish the configuration files for the central administration from the normal rvs® Tiny XML configuration files such as `rvsTinyStationlist.xml`, the files for the central administration will be named **configuration container files**.

These are

– `cfg.req.jar` for configuration requests, sent by LCI to RCI and
– `cfg.rsp.jar` for configuration responses, sent by RCI to LCI.

The Local Configuration Instance LCI

Each rvs® Tiny installation can be used as LCI when during the setup procedure in the window „Choose Product Feature“ the „Complete Installation“ is selected (please see the chapter 2.3 for the explanation how to install rvs® Tiny).

© gedas deutschland GmbH / Fachbereich rvs® Systems

The LCI maintains a special directory where the configuration data of all RCI's (rvs® Tiny stations, that are to be administrated) are stored - the Configuration Repository CRep. The location of CRep is `$RVSTINY_HOME/management`. Please refer to the chapter 1.2 for the explanation of `$RVSTINY_HOME`.

The entries of CRep (`$RVSTINY_HOME/management`) are:

– `$RVSTINY_HOME/management/mgmt-datastore`
This directory will be created only after a succesfull transmission and response to the request for the configuration file of the RCI. Please see the chapter 5.2 for the explanation how to make a configuration request.

– `$RVSTINY_HOME/management/mgmt-log/activity-log`
This log file contains protocols of all configuration actions. Each entry comprises a timestamp, the type: **configuration request** or **configuration response** respectively, the SID of the administrated rvs® Tiny station (RCI) and a message text.

– `$RVSTINY_HOME/management/mgmt-templates`
This directory contains templates for management actions. It is not necessary to deal with this directory except for updates of the administration software itself.

– `$RVSTINY_HOME/management/mgmt-workspace`
This directory stores configurations of RCIs for editing. It contains subfolders, that are named after the RCI's SID (see the tool `prepareUpdateStation`, chapter 5.2 for an example).

The Remote Controlled Instance RCI

The RCI is a is rvs® Tiny station, which should be administrated remotely. It can be a „Typical Installation" of rvs® Tiny (Please see the chapter 2.3 for the explanation of the installation procedure) and does not have to be a „Complete Installation" such as LCI.

When rvs® Tiny (RCI) receives a configuration request (a file `cfg.req.jar`) a dedicated job will be launched. This job starts the rvs® Tiny configuration process that handles the configuration request and generates the configuration response. The configuration response is sent back from the RCI to LCI as file `cfg.rsp.jar`.

5.2 Command Tools of the Central Administration

The whole process (all configuration and administration cases) is to be done by the following command tools. These tools are located in the directory `$RVSTINY_HOME\bin` as batch files (please see the chapter 1.2 for the explanation of `$RVSTINY_HOME`.), so you must change to this directory to be able to execute them.

orderConfiguration.bat -s SID	fetches the configuration of an RCI and stores it in CRep. **Example**: `C:\rvsTiny\management\mgmt-datastore\TINYPW` In this example `TINYPW` is the SID of RCI (of the station, that should be administrated).
prepareUpdateStation.bat -s <SID>	gets a RCI configuration copy out of CRep to the WorkDir in order to serve as starting point for modifications. **Example** for the WorkDir: In the directory `C:\rvsTiny\management\mgmt-workspace` the following subdirectory will be created `TINYPW\UPDATE_STATION_040826_114418\out.` This is a subdirectory for the `TINYPW` station; the date of the creation is 2004-08-26, the time of the creation is 11:44:18.
commitUpdateStation.bat -s <SID> -d <WorkDir>	sends a modified configuration from the WorkDir (without `out` subdirectory) to an RCI (RCI should be set with the option -s for stationID). **Example**: `commitUpdateStation.bat -s TINYPW -d C:\rvsTiny\management\mgmt-workspace\TINYPW\UPDATE_STATION_040826_114418`

Note: The name of the WorkDir directory consists of CRep (see 5.1); RCI'SID (`TINYPW` in the table example), directory `UPDATE_STATION` with the timestamp (consisting of date and time; in the table example `040826_114418`) and the directory `out`.

WorkDir contains all essential rvs® Tiny directories and files: By changing these files the following configuration actions for example are possible:

– Modify station configuration by modifying `$RVSTINY_HOME/conf/rvsTinyStationlist.xml`.

– Modify jobstarts by modifying `$RVSTINY_HOME/conf/rvsTinyJobstart.xml`.

– Update the software by replacing .jar files in the directory `$RVSTINY_HOME/lib` .

© gedas deutschland GmbH / Fachbereich rvs® Systems

5.3 How to work with the central administration features?

The following steps will show you the typical configuration run. It may be used as a basic example when a rvs® Tiny network administrator deals with rvs® Tiny Central Administration. These steps are always necessary, independent of the fact, whether you want to make an update, change some rvs® Tiny parameter or exchange the license key file. More details about the particular administation tasks, you will find in the chapters 5.3.1, 5.3.2 and 5.3.3.

Note: The following commands are available as batch files in the `$RVSTINY_HOME\bin` directory (please see the chapter 1.2 for the explanation of `$RVSTINY_HOME`.), so you must change to this directory to be able to execute them.

- at first, get the configuration of the RCI (of the rvs® Tiny station, that should be administrated). Use the program `orderConfiguration` for this step.

  Example: If you would like to administer the configuration of the rvs® Tiny station `TINY11`, launch the program `orderConfiguration` in the command line with the following command:

  ```
  orderConfiguration -s TINY11
  ```

  This command generates a configuration request file `cfg.req.jar` (see the chapter 5.1 for the explanation of the meaning of `cfg.req.jar`) and sends it via OFTP to the station `TINY11`. The transmission of the file `cfg.req.jar` may be watched in the rvs® Tiny GUI (Admin window). When the RCI (`TINY11`) receives the file `cfg.req.jar` the configuration process will be started. The configuration process stops rvs® Tiny (station `TINY11`), archives the actual configuration in a file `cfg.rsp.jar`, starts rvs® Tiny (`TINY11`) again and sends the configuration container file `cfg.rsp.jar` back to the LCI (rvs® Tiny station of the administrator). If this step was successful, you will receive a message „OrderConfiguration exited with return code *0*“ in the console. All these steps are part of the program `orderConfiguration` and will be executed automatically.

  The next step is to get a copy of the RCI's configuration, that arrived as the file `cfg.rsp.jar`. This should be done by the program `prepareUpdateStation` on the command line. This program will copy for you the arrived configuration file `cfg.rsp.jar` and store it to the WorkDir (see chapter 5.2 for the explanation of WorkDir).

  Example:

  ```
  prepareUpdateStation.bat -s TINY11
  ```

  Result: The directory `C:\rvsTiny\management\mgmt-workspace\TINY11\UPDATE_STATION_040828_113315\out` with the complete configuration of the station `TINY11` will be created. If this action was successfull, you will find a corresponding message in the file `activity.log`. This log file is stored in the directory `$RVSTINY_HOME/management/mgmt-log`.

- Now you can administrate the configuration of the rvs® Tiny station (e.g. `TINY11`). You can exchange a license key, modify the XML configuration files or substitute the appropriate .jar files to make an update of rvs® Tiny. Please read the chapters 5.3.1, 5.3.2 und 5.3.3 for more details about the particular configuration procedures.
- Send the modified configuration to the RCI (`TINY11`). You have to send the whole directory `C:\rvsTiny\management\mgmt-workspace\TINY11\UPDATE_STATION_040828_113315)` with the command:

  ```
  commitUpdateStation.bat -s TINY11 -d
  C:\rvsTiny\management\mgmt-
  workspace\TINY11\UPDATE_STATION_040828_113315
  ```

  This command will store the whole modified directory and send it to the RCI (`TINY11`) again as a file `cfg.req.jar`.

  After successfully receiving a file `cfg.req.jar` at the RCI (`TINY11`), rvs® Tiny will be stopped (it all happens with the process `commitUpdateStation`, you do not have to do any particular steps); the modified configuration will be updated; the update job checks, if all was correct and sends back a response as a file `cfg.rsp.jar`. The result of the update is again logged in the file `activity.log`.

Note: In case of non success the old configuration will be activated again.

5.3.1 How to exchange a license key file?

This chapter describes the typical case in administrating rvs® Tiny, how to exchange an invalid license key. In this example the station `TINY01` will administrate the station `TINY02`.

Prerequisites: The station `TINY01` must have the station `TINY02` in the station table as routed station via the rvs® center (please read the chapter 3.2.1 for the explanation how to set up stations) and the station `TINY02` must also have station the `TINY01` as routed station via the rvs® center. The station `TINY01` must be a „Complete Installation“ and the station `TINY02` only a „Typical Installation“ (please see the chapter 2.3 for the explanation of „Typical“ and „Complete Installation“).

- To be able to replace a license key file of `TINY02`, `TINY01` must at first get the configuration of `TINY02` with the command:
  `orderConfiguration -s TINY02`

  If this step was successful, you will receive a message „OrderConfiguration exited with return code *0*“ in the console and the file `cfg.rsp.jar` will be received. (see in the **Ended Transmissions**, Admin-window of the `TINY01` GUI).
- The next step ist to get a copy of the `TINY02` configuration, that arrived as the file `cfg.rsp.jar.` This must be done with the following command:

  ```
  prepareUpdateStation.bat -s TINY02
  ```


© gedas deutschland GmbH / Fachbereich rvs® Systems

Result: The directory `C:\rvsTiny\management\mgmt-workspace\TINY02\UPDATE_STATION_040828_113315\out` with the complete configuration of the station `TINY02` will be created. If this action was successful, you will find the message in the file `activity.log`. This log file is stored in the directory `$RVSTINY_HOME/management/mgmt-log`.

- Now you can rename the old license key from the directory `C:\rvsTiny\management\mgmt-workspace\TINY02\UPDATE_STATION_040828_113315\out\conf` to `licenseOLD.properties` and copy the new license key `license.properties` to the directory `C:\rvsTiny\management\mgmt-workspace\TINY02\UPDATE_STATION_040828_113315\out\conf`. How to obtain the new license key, please read the chapter 2.2.
- Send the modified configuration to `TINY02` (you must send the whole directory `C:\rvsTiny\management\mgmt-workspace\TINY02\UPDATE_STATION_040828_113315)` with the command

  ```
  commitUpdateStation.bat -s TINY02 -d
  C:\rvsTiny\management\mgmt-
  workspace\TINY02\UPDATE_STATION_040828_113315
  ```

  This command will store the whole modified directory and send it to the station `TINY02` again as a file `cfg.req.jar`.
- After successfully receiving the file `cfg.req.jar` at the station `TINY02`, rvs® Tiny at `TINY02` will be stopped (it happens all with the process `commitUpdateStation`, you do not have to do any particular steps); the modified configuration will be updated; the update job checks, if all was correct and sends back a response as a file `cfg.rsp.jar`. The result of the update is again logged in the file `activity.log`.

Note: In case of non success the old configuration will be activated again.

5.3.2 How to change a station parameter?

This chapter describes a typical case in administrating rvs® Tiny, how to change a rvs® Tiny parameter e.g. the ODETTE-ID. It is the same procedure for changing any other rvs® Tiny parameter. In this example the station `TINY20` will administrate the station `TINY22`.

Prerequisites: The station `TINY20` must have the station `TINY22` in the station table as routed station via the rvs® center (please read the chapter 3.2.1 for explanation how to set up stations) and the station `TINY22` must also have station `TINY20` as routed station via the rvs® center. The station `TINY20` must be a „Complete Installation“ and the station `TINY22` only a „Typical Installation“ (please see the chapter 2.3 for the explanation of „Typical“ and „Complete Installation“).

- To be able to modify an rvs® Tiny parameter `TINY20` must at first get the configuration of `TINY22`. it will be done with the following command:

  ```
  orderConfiguration -s TINY22
  ```

  If this step was successful, you will receive a message „OrderConfiguration exited with return code *0*“ in the console and the file `cfg.rsp.jar` from the station `TINY22` will be received as a response of this request; please see in the **Ended Transmissions**, Admin-window of the `TINY20` GUI.

- The next step ist to get a copy of the `TINY22` configuration, that arrived as the file `cfg.rsp.jar` to the WorkDir of `TINY20`. Please see the chapter 5.2 for the explanation of the WorkDir.

  ```
  prepareUpdateStation.bat -s TINY22
  ```

  Result: The directory `C:\rvsTiny\management\mgmt-workspace\TINY22\UPDATE_STATION_040829_133315\out` with the complete configuration of the station `TINY22` will be created. If this action was successful, you will find a corresponding message in the file `activity.log`. This log file is stored in the directory `$RVSTINY_HOME/management/mgmt-log`.

- Now you can edit the file `rvsTinyStationlist.xml` from the directory `C:\rvsTiny\management\mgmt-workspace\TINY22\UPDATE_STATION_040829_133315\out\conf` and modify it e.g. parameter ODETTE_ID or TCP/IP address (parameter IP_ADDR) for the local station of `TINY22` (STATION_LOC) or other stations (rvs® center or routed stations).

- The next step is to send the modified configuration to `TINY22` (you must send the whole directory (but without `out` directory) `C:\rvsTiny\management\mgmt-workspace\TINY02\UPDATE_STATION_040829_133315)` with the command

  ```
  commitUpdateStation.bat -s TINY22 -d
  C:\rvsTiny\management\mgmt-
  workspace\TINY22\UPDATE_STATION_040829_133315
  ```

  This command will store the whole modified directory and send it to station `TINY22` again as a file `cfg.req.jar`.

- After successfully receiving a file `cfg.req.jar` at the station `TINY22`, rvs® Tiny at station `TINY22` will be stopped (it all happens with the process `commitUpdateStation`, you do not have to make any particular steps) and the modified configuration will be updated. Then the update job checks, if all was correct and sends back a response to the station `TINY20` as a file `cfg.rsp.jar`. The result of the update is again logged in the file `activity.log`.

Note: In case of non success the old configuration will be activated again.

© gedas deutschland GmbH / Fachbereich rvs® Systems

5.3.3 How to make an update of rvs® Tiny?

This chapter describes a typical case in administrating rvs® Tiny, how to make an update of an another rvs® Tiny station. In this example the station `TINY30` will administrate the station `TINY33`.

Prerequisites: The station `TINY30` must have the station `TINY33` in the station table as routed station via the the rvs® center (please read the chapter 3.2.1 for explanation how to set up stations) and the station `TINY30` must also have station `TINY33` as routed station via the rvs® center. The station `TINY30` must be a „Complete Installation“ and the station `TINY33` only a „Typical Installation“ (please see the chapter 2.3 for the explanation of „Typical“ and „Complete Installation“).

- To be able to make an update of rvs® Tiny at the station `TINY30` you must get at first the configuration of `TINY33`.

  ```
  orderConfiguration -s TINY33
  ```

  If this step was successful, you will receive a message „OrderConfiguration exited with return code 0“ and the file `cfg.rsp.jar` will be received, see in the **Ended Transmissions**, Admin-window of the `TINY30` GUI.

- The next step ist to get a copy to the WorkDir of the `TINY33` configuration, that arrived as the file `cfg.rsp.jar`. Type the follwing command in the command line:

  ```
  prepareUpdateStation.bat -s TINY33
  ```

  Result: The directory `C:\rvsTiny\management\mgmt-workspace\TINY33\UPDATE_STATION_040830_113315\out` with the complete configuration of the station `TINY33` will be created. If this action was successful, you will find a corresponding message in the file `activity.log`.

- Now you must rename the old .jar file `rvstiny.jar` from the directory `C:\rvsTiny\management\mgmt-workspace\TINY33\UPDATE_STATION_040830_113315\out\lib` to `rvstinyOLD.jar` and replace it with the new one. Please contact us to receive the actual files for the update (E-Mail: rvs-service@gedas.de; Tel. +49 30 39971 777; fax: +49 30 39971 994).

- Send the modified configuration to `TINY33` station (you must send the whole directory `C:\rvsTiny\management\mgmt-workspace\TINY33\UPDATE_STATION_040830_113315`) with the command

  ```
  commitUpdateStation.bat -s TINY33 -d
  C:\rvsTiny\management\mgmt-
  workspace\TINY02\UPDATE_STATION_040830_113315
  ```

  This command will store the whole modified directory and send it to `TINY33` again as a file `cfg.req.jar`.

- After successfully receiving the file `cfg.req.jar` at the station `TINY33`, rvs® Tiny of `TINY33` will be stopped (it all happens with the process `commitUpdateStation`, you do not have to do any particular steps); the modified configuration will be updated; the update job checks, if all was correct and sends back a response as a file `cfg.rsp.jar`. The result of the update is again logged in the file `activity.log`.

Note: In case of non success the old configuration will be activated again.

5.4 How to change from rvsWIN to rvs® Tiny?

In this chapter are described the special circumstances for the user, who would like to change from rvsWIN to rvs® Tiny .

rvsWIN is a File-Transfer-Application under MS-Windows in connection with rvsMVS. It is not based on OFTP (Odette File Transfer Protocol). It is an 3270 emulation with a SNA-Line.

rvs® Tiny is an OFTP based file transfer solution for large star topology networks comprising several hundred remote stations. It is implemented in Java; please read the chapter 2.1 about the system requirements (JRE 1.3.1_XX or Java Software Development Kit 1.3.1_XX).

The following steps are necessary, if you want to change from rvsWIN to rvs® Tiny:

– the new rvs® Tiny installation setup program

– the valid license key (file `license.properties`). Please read the chapter 2.2 for the explanation how to obtain the rvs® Tiny license.

– In the installation procedure of rvs® Tiny are some mandatory parameters, that didn't exist in rvsWIN. These are:

  ODETTE-ID (please refer to chapter 3.2.3 for the explanation how to obtain this ID),

  TCP/IP Address (or the hostname): it is the IP address of your computer. If this address is not known to you, type in the command line the command `hostname` (for the host name) or the command `ipconfig` for the IP address of your computer.

  Passwords: receive and send passwords should be exchanged with the direct neighbour station (rvs® center) in bilateral agreement. Your send password is the receive password at the rvs® center station. The send password of the rvs® center is your receive password. For routed stations you do not need passwords.

– For rvs® Tiny you do not have to be a RACF user, as it is necessary for rvsWIN, it is the ODETTE-ID, which identifies the participients in the file transfer.

Some parallels between rvs® Tiny and rvsWIN:

– rvs® Tiny, as well as rvsWIN enables to send files in format fixed or variable to the mainframe computer (e.g. rvsMVS).

© gedas deutschland GmbH / Fachbereich rvs® Systems

- Code translation (code conversion) ASCII -> EBCDIC when sending or receiving files: also possible in rvs® Tiny, but with a greater number of selectable conversion tables.
- Resident Receive Entries in rvsWIN are equivalent to Jobstarts after receive in rvs® Tiny (please read the chapter 3.3 of rvs® Tiny User Manual for more information)
- The Control File of rvsWIN is equivalent to Transfer window, directory Jobs in the rvs® Tiny GUI (please see the chapter 4.7).

A
activateStation *36*
ARCDIR *18*
archiveJobs *51*

B
Batch file *33*

C
ConfigFile *17*
Configuration files *29*
CONTACT *24*, *28*
createSendJob *38*
Customizing configuration files *17*

D
DB *18*

E
EERP *25*, *29*
EERP_OUT *25*, *28*, *29*, *50*
ENDED *18*, *42*
ENTERPRISE *25*
EX_BUF_CRE *27*
EX_BUF_SIZ *27*

F
FAILED *18*, *42*

G
GATEWAY_STATION_NK *28*
getJob *49*
getJobList *48*

H
handleEERP *25*, *28*, *29*, *50*
HOLD *25*, *28*, *29*
HostAllowFile *18*
HostDenyFil *18*

I
IMMEDIATE *25*, *28*, *29*
INBOX *18*
IP_ADDR *25*, *28*, *29*

J
jobFilter *29*
JobStart *29*
JobstartConfigFile *18*

L
LINE_TYPE *27*
LOCATION *25*
LogConfigFil *17*
LOGDIR *18*

M
monlog.log *18*

N
NAME *25*

O
ODETTE *26*, *27*, *28*
Odette port *24*
ODETTE_ID *24*, *27*, *28*
OUTBOX *18*

P
Platforms *7*
Port *25*, *28*, *29*

R
RCV *18*
REC_PW *25*, *27*
revision.log *51*
RMIServiceHost *19*
RMISeviceName *19*
RootDir *17*
Routing station *28*
rvs® center *26*
rvsTiny.properties *17*
$RVSTINY_HOME *8*
rvsTinyConfig.xml *18*
rvsTinyStationlist.xml *19*

S
sendAttempts *30*
showMonitorLog *35*
SID *24*, *26*, *27*, *28*
SN *18*
SND_PW *25*, *27*
startServer *33*

© gedas deutschland GmbH / Fachbereich rvs® Systems

STATION_LOC *26*
STATION_NK *26*
STATION_RT *28*
StationsConfigFile *18*
stopServer *33*
STREET *25*
STREETNUMBER *25*

T

TCPIP_BASIC *24*, *25*, *28*, *29*
TCPIP_REC *25*, *26*
TEMP *18*
tiny.log *18*
Typographic conventions *8*

V

VDA *25*
VDSN *30*
vdsn *30*

W

What is rvs® *5*
What rvs® is not *5*